IMPLEMENTASI *SYSTEM REAL TIME* UNTUK *MONITORING* PENCAHAYAAN SUHU DAN KELEMBABAN PADA TANAMAN STROBERI

SKRIPSI

KEMINATAN TEKNIK KOMPUTER

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Agung Leona Suparlin
NIM: 135150301111015

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI *SYSTEM REAL TIME* UNTUK *MONITORING* PENCAHAYAAN SUHU DAN KELEMBABAN PADA TANAMAN STROBERI

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Agung Leona Suparlin
NIM: 135150301111015

Skripsi ini telah diuji dan dinyatakan lulus pada
16 Januari 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T, M.Eng	Dahnial Syauqy, S.T, M.T, M.Sc
NIP: 19820809 201212 1 004	NIK: 20160787 042310 02

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

IDENTITAS PENGUJI

Tibyani, S.T, M.T (ke 1) *Ketua Majelis

NIP. 19691101 199512 1 002

Wijaya Kurniawan, S.T, M.T (ke 2)

NIP. 19820125 201504 1 002

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 16 Januari 2018

Agung Leona Suparlin

NIM: 135150301111015

DAFTAR RIWAYAT HIDUP

Nama	: Agung Leona Suparlin
Tempat, Tanggal Lahir	: Bandung, 11 Agustus 1995
Jenis Kelamin	: Laki-laki
Agama	: Islam
Alamat	: Kp. Juntigirang RT 01/Rw 12 Desa Banyusari Kecamatan Katapang Kabupaten Bandung

Riwayat Pendidikan

SDN Juntigirang IV	Tahun Lulus 2007
SMPN 2 Katapang	Tahun Lulus 2010
SMK Angkasa 1 Margahayu	Tahun Lulus 2013

UCAPAN TERIMAKASIH

1. Kedua Orang Tua, Bapak Parman dan Ibu Marlina Lapenia serta seluruh keluarga besar atas segala nasehat, kasih sayang dan kesabaran dalam membesarkan dan mendidik peneliti, serta senantiasa tiada hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
2. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Heru Nurwarsito, Ir., M. Kom. selaku Wakil Dekan I Bidang Akademik Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang..
5. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. selaku Ketua Program Studi Teknik Komputer Universitas Brawijaya Malang.
6. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. selaku dosen pembimbing I yang telah memberikan pengarahan dalam pembuatan skripsi ini.
7. Bapak Dahnial Syauqy, S.T, M.T, M.Sc. selaku dosen pembimbing II yang telah memberikan pengarahan dalam pembuatan laporan skripsi ini.
8. Seluruh civitas akademika Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penyelesaian skripsi ini.
9. Kepada calon istri peneliti, Hasna Nur Zakiyah Darajat yang selalu mendukung dan memberi motivasi untuk kelancaran skripsi ini.
10. Kepada Capaba Family, Hafizh Hamzah Wicaksono S.kom dan Mohammad Misfaul May Dana S.kom yang dalam pengerjaannya alat saling membantu sekaligus bertukar pendapat.
11. Oyisam Squad, Duwi Hariyanto, Fajar Andika, Raihan Al-Hakim, Nur Fauzi S.kom, Septian Mukti S.kom, Puguh Bahtiar S.kom yang memberikan dukungan dan hiburan kepada peneliti.
12. Teman-teman Teknik Komputer angkatan 2013 yang selalu mendukung dan berbagi ilmu dari awal perkuliahan sampai tahap akhir penyelesaian skripsi dan semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat penulis sebutkan satu per satu.

ABSTRAK

Fragaria adalah genus tumbuhan berbunga dalam keluarga mawar, Rosaceae, yang dikenal secara umum dengan nama stroberi karena buahnya yang bisa dikonsumsi. Akan tetapi, Ada beberapa faktor yang perlu diperhatikan agar tanaman stroberi dapat tumbuh dengan baik yaitu batasan nilai untuk pencahayaan, suhu, dan kelembaban tanah. Jika faktor tersebut tidak terpenuhi, maka pertumbuhan tanaman stroberi tidak akan maksimal ataupun tanaman akan layu dan mati. Berdasarkan permasalahan tersebut, diperlukan sebuah sistem yang mampu melakukan pemantauan berdasarkan pencahayaan, suhu dan kelembaban tanah yang baik pada tanaman stroberi. Pada tugas akhir terdapat 3 sensor yaitu sensor cahaya LDR, sensor suhu LM35, dan sensor kelembaban tanah SEN0114 yang terhubung dengan microcontroller Arduino mega. Output dari sistem ditampilkan menggunakan LCD dan lampu peringatan LED RGB. Pada arduino mega ditanamkan RTOS yang bertugas sebagai penjadwalan task pada sistem. RTOS yang digunakan sudah disediakan pada library Arduino Mega yaitu FreeRTOS. Dari hasil pengujian, sistem dapat menentukan berbagai kondisi pada pemantauan tanaman stroberi dengan keakuratan mencapai 100%. Penjadwalan task yang dilakukan sesuai dengan prioritas yang dibuat. Rata-rata waktu eksekusi sistem dengan menggunakan RTOS adalah 88,4ms, sedangkan sistem tanpa RTOS adalah 76,9ms. Sistem dengan RTOS memerlukan waktu yang lebih lama, dikarenakan pada sistem tersebut terdapat fungsi take-and-give semaphore yang membutukan waktu eksekusi selama ±1,65ms.

Kata kunci: Stroberi, Pemantauan, RTOS, Prioritas, FreeRTOS

ABSTRACT

Fragaria is a genus of flowering plants in family rose, rosaceae, known in general in the name of strawberries for its fruit that can be consumed.But, there are several factors that it should be noted that plant strawberries can grow well the limit values for lighting, temperature, and humidity land.If factors are not achieved, but the growth of plants strawberries will not maximum or plant will dry up and died. Based on these problems, required a system that can afford to run monitoring based on lighting, temperatures and moisture land in plants strawberries. In this research there are 3 sensors that is the censor light LDR, sensors temperature LM35, and sensors moisture land SEN0114 connected with microcontroller arduino mega.The output of the system displayed using LCD and the warning lights LED RGB. In arduino mega embedded RTOS that served as scheduling task on the system. The RTOS used is already provided in the Arduino Mega library ie FreeRTOS. From the test results, the system can be determine the range of conditions in monitoring the strawberries accuracy reaches 100 %. Scheduling tasks performed in accordance with the priorities made. The average system execution time using RTOS is 88,6ms, while the system without RTOS is 76,9ms. System with RTOS takes longer time, because in system there is take-and-give semaphore function which need execution time for ± 1,64ms.

Keywords: Strawberrie, Monitoring, RTOS, Priority, FreeRTOS

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Tuhan Yang Maha Esa, karena Rahmat-Nya penulis dapat menyelesaikan skripsi ini. Adapun maksud penyusunan skripsi ini adalah untuk memenuhi salah satu persyaratan dalam menempuh ujian Sarjana Fakultas Ilmu Komputer. Judul skripsi yang disusun adalah:" Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi".

Banyak kesulitan dan hambatan yang dialami oleh penulis dalam menyusun skripsi ini terutama dalam mendapatkan data dan mengolahnya, tetapi semua itu telah dapat diatasi dengan baik berkat dukungan dan bantuan dari berbagai pihak. Berdasarkan hal tersebut, pada kesempatan ini penulis mengucapkan terima kasih kepada yang terhormat:

1. Kedua Orang Tua, Bapak Parman dan Ibu Marlina Lapenia serta seluruh keluarga besar atas segala nasehat, kasih sayang dan kesabaran dalam membesarkan dan mendidik peneliti, serta senantiasa tiada hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
2. Bapak Wayan Firdaus Mahmudy, S.Si, M.T, Ph.D. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya Malang.
3. Bapak Heru Nurwarsito, Ir., M. Kom. selaku Wakil Dekan I Bidang Akademik Fakultas Ilmu Komputer Universitas Brawijaya Malang.
4. Bapak Tri Astoto Kurniawan, S.T, M.T, Ph.D selaku Ketua Jurusan Teknik Informatika Fakultas Ilmu Komputer Universitas Brawijaya Malang..
5. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. selaku Ketua Program Studi Teknik Komputer Universitas Brawijaya Malang.
6. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng. selaku dosen pembimbing I yang telah memberikan pengarahan dalam pembuatan skripsi ini.
7. Bapak Dahnial Syauqy, S.T, M.T, M.Sc. selaku dosen pembimbing II yang telah memberikan pengarahan dalam pembuatan laporan skripsi ini.
8. Seluruh civitas akademika Fakultas Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama penyelesaian skripsi ini.
9. Kepada calon istri peneliti, Hasna Nur Zakiyah Darajat yang selalu mendukung dan memberi motivasi untuk kelancaran skripsi ini.
10. Kepada Capaba Family, Hafizh Hamzah Wicaksono S.kom dan Mohammad Misfaul May Dana S.kom yang dalam pengerjaannya alat saling membantu sekaligus bertukar pendapat.
11. Oyisam Squad, Duwi Hariyanto, Fajar Andika, Raihan Al-Hakim, Nur Fauzi S.kom, Septian Mukti S.kom, Puguh Bahtiar S.kom yang memberikan dukungan dan hiburan kepada peneliti.
12. Teman-teman Teknik Komputer angkatan 2013 yang selalu mendukung dan berbagi ilmu dari awal perkuliahan sampai tahap akhir penyelesaian skripsi dan semua pihak yang telah membantu terselesaikannya skripsi ini yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari bahwa dalam penyusunan skripsi ini masih jauh dari sempurna, sehingga untuk segala saran dan kritik yang membangun penulis ucapkan terimakasih. Akhir kata penulis berharap skripsi ini dapat membawa manfaat bagi semua pihak yang menggunakannya.

Malang, 16 Januari 2018

Penulis
agungleona@gmail.com

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Perkotaan identik dengan banyaknya gedung-gedung bertingkat dan jadwal aktivitas masyarakat yang padat, bagi masyarakat yang tinggal di perkotaan sulit menemukan lahan kosong yang luas dan masalah keterbatasan waktu untuk melakukan kegiatan bercocok tanam ataupun hanya sekadar menyalurkan hobi untuk mengoleksi tanaman (Ramdani, 2015). Pemanfaatan pot dapat dijadikan solusi untuk melakukan kegiatan bercocok tanam atau mengoleksi tanaman. Selain itu perlu diperhatikan tentang faktor-faktor yang mempengaruhi pertumbuhan agar tanaman dapat tumbuh dengan baik.

Pada penelitian yang dilakukan, objek yang digunakan adalah tanaman stroberi, selain pembudidayaan stroberi yang menjajikan di pasar produksi, tanaman ini juga cocok di implementasikan ke dalam pot karena memiliki tangkai dan dahan yang pendek. Stroberi (Fragaria sp.) merupakan tanaman buah yang di temukan pertama kali di Chili yaitu spesies Fragaria chiloensis L. yang menyebar ke berbagai negara Amerika, Eropa dan Asia; sedangkan Fragaria vesca L. yang berasal dari Amerika Utara lebih menyebar luas dan jenis rendah, mengandung vitamin C, asam folat, kalium dan antioksidan yang tinggi. Biji dan daun stroberi mengandung asam elegat yang bermanfaat untuk mengurangi resiko terserang kanker (Kurnia, 2005).

Produksi stroberi dunia mengalami peningkatan setiap tahunnya. Dan Negara yang paling banyak menghasilkan dan mengkonsumsi buah stroberi adalah Amerika Serikat. Dapat di lihat produksi buah stroberi di Amerika antara 2005-2007 mengalami peningkatan produksi. Pada tahun 2005 produksinya 1,053,242ton/ ha, dan meningkat menjadi 1,090,436 ton/ha pada tahun 2006 dan terus meningkat menjadi 1,115,000 ton/ha pada tahun 2007. Dari data tersebut dapat dilihat bahwa prospek agribisnis stroberi cukup cerah dilihat dari daya serap pasar dan permintaan dunia dari tahun ke tahun meningkat. Prospek usaha stroberi sangat menjanjikan, produksi buah yang sampai sekarang belum dapat memenuhi permintaan pasar ini memiliki harga jualyang cukup tinggi. Produk olahan stroberi juga banyak diminati di pasaran, stroberi juga dapat diolah menjadi selai, manisan, sirup, dodol, yoghurt, maupun es krim. (Cahyono, 2011)

Tanaman ini tumbuh dengan tangkai yang relatif pendek, maka pemanfaatan pot sangat cocok untuk di implementasikan pada tanaman stroberi. Ada beberapa faktor yang pelu diperhatikan agar tanaman stroberi dapat tumbuh dengan baik, yaitu tanaman stroberi dapat tumbuh baik di daerah dengan curah hujan 600-700 mm/tahun, kemudian penyinaran cahaya matahari dalam pertumbuha adalah 8–10 jam setiap harinya, lalu kelembaban tanah yang ideal untuk budidaya stroberi adalah 80-90%. Stroberi adalah tanaman subtropis yang dapat beradaptasi dengan baikdi dataran tinggi tropis yg memiliki temperatur 17–20˚C. Jika faktor-faktor tersebut tidak terpenuhi, maka

pertumbuhan tanaman stroberi tidak akan maksimal ataupun tanaman akan layu dan mati (Hermawan, 2016). Oleh karena itu, dibutuhkan suatu sistem atau alat yang dapat membatu pengguna dalam memantau pertumbuhan tanaman stroberi agar beberapa faktor yang mempengaruhi pertumbuhan tanaman dapat terpenuhi sehingga tanaman dapat tumbuh secara maksimal dan menghasilkan buah dengan kualitas baik.

Pemantauan adalah suatu proses pengumpulan danmenganalisis informasi dari penerapan suatu program termasuk mengecek secara reguler untuk melihat apakah program itu berjalan sesuai rencana sehingga masalah yang ditemui dapat diatasi (Kumorotomo, 2003). Pada *embedded system*, sistem operasi sangat diperlukan dalam pengaturan eksekusi yang menuntut kecepatan proses. Selain dibutuhkan sistem yang merespon dengan cepat perubahan masukkan, melakukan proses multi*task*ing, dan menjamin ketepatan hasil eksekusi, diperlukan juga sistem yang memiliki kepastian waktu selesainya sebuah pekerjaaan atau *task*. *Real-Time Operating System* (RTOS) merupakan salah satu solusi yang sesuai untuk mengaplikasikannya pada embedded mikrokontroler (Jatmiko, 2015).

FreeRTOS digunakan untuk mewujudkan *system* berjalan secara *real time*. Ahn dan Tan melakukan survey dan mengamati performa dari beberapa RTOS yang beredar di pasaran. Dalam hasil penelitiannya menyatakan bahwa RTOS sangat baik untuk di implementasikan dalam sebuah *embedded system* di bandingkan dengan *generic OS*. Hal tersebut di karenakan beberapa factor seperti *preemptive, priority-based scheduling, predictability in task synchronization, deterministic behavior*. Perbedaan utama yang perlu diperhatikan antara RTOS dengan *generic OS* adalah jika pada RTOS waktu harus bisa diprediksi 0dan tetap konsisten walaupun tugas yang dikerjakan bertambah sedangkan pada generic OS tidak dapat melakukan hal tersebut (Anh & Tan, 2009).

Dari pemaparan diatas, penulis ingin merancang sebuah *system monitoring* dengan *real time system* (sistem waktu nyata) dengan judul "Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi". Penulis berharap hasil penelitianya ini dapat membantu para produsen buah stroberi untuk memantau proses pembudidayaan agar tanaman dapat tumbuh dengan maksimal dan menghasilkan buah dengan kualitas baik.

1.2 Rumusan masalah

Berdasarkan pada permasalahan yang telah diangkat pada bagian latar belakang. Adapun rumusan masalah dalam penelitian ini adalah :

1. Bagaimana proses akusisi data dari sensor cahaya LDR, sensor suhu LM35 dan sensor kelembaban tanah SEN0114?
2. Bagaimana pengaruh *prioritas task* pada RTOS?
3. Bagaimana pengaruh RTOS jika ditinjau dari waktu eksekusi?
4. Bagaimana perbedaan waktu eksekusi sistem yang menggunakan RTOS dengan sistem yang tanpa menggunakan RTOS?

1.3 Manfaat

Manfaat yang diharapkan dari penelitian ini adalah sebagai berikut.

1. Bagi Penulis :
    a) Dapat memperdalam pengetahuan tentang pembudidayaan tanaman stroberi.
    b) Mengetahui cara kerja RTOS pada sistem yang dibuat.
2. Bagi Pengguna:
    a) Dapat membantu para produsen buah stroberi untuk memantau proses pembudidayaan agar tanaman dapat tumbuh dengan maksimal dan menghasilkan buah dengan kualitas baik.
3. Bagi Pembimbing:
    a) Dapat menjadi media dalam pembelajaran.

1.4 Batasan masalah

Dalam melakukan penelitian ini, diperlukan batasan masalah agar pengerjaannya dapat terarah dan tidak terlalu meluas. Maka perlu diterapkan batasan permasalahan yaitu Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi. yang akan dibahas adalah sebagai berikut :

1. Objek yang menjadi penelitian adalah buah stroberi yang tertanam pada pot.
2. Yang akan dijadikan objek pemantauan adalah paparan cahaya, suhu dan kelembaban tanah.
3. RTOS yang digunakan dalam sistem ini adalah FreeRTOS.
4. FreeRTOS bertugas sebagai penjadwal *task* sistem
5. Output pada sistem ini berupa tampilan akuisisi data pada suhu, pencahayaan dan kelembaban serta peringatan yang ditandai dengan nyala lampu LED.

1.5 Sistematika pembahasan

Sistematika penulisan ini memberikan gambaran dan penjelasan secara garis besar dari seluruh isi penelitian yang terdiri dari beberapa bab, yang terdiri dari:

BAB 1 Pendahuluan

Pada bab ini menjelaskan tentang latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah, dan sistematika pembahasan mengenai “Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi”.

BAB 2 Landasan Kepustakaan

Pada bab ini menjelaskan tentang dasar teori, dan kajian pustaka dari metode yang digunakan dalam penelitian dengan judul "Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi"

BAB 3 Metode Penelitian

Pada bab ini membahas tentang langkah kerja dan metode yang digunakan dalam penelitian, dengan beberapa proses yaitu studi literatur, analisis kebutuhan, perancangan sistem, implementasi sistem, pengujian sistem, dan penarikan kesimpulan.

BAB 4 Rekayasa kebutuhan

Bab ini menguraikan tentang kebutuhan-kebutuhan yang diperlukan untuk merancangan sistem *monitoring* pada suhu, pencahayaan dan kelembaban pada tanaman stroberi, seperti kebutuhan fungsional sistem, perangkat keras, dan perangkat lunak.

BAB 5 Perancangan dan implementasi

Pada bab ini membahas tentang implementasi dari penelitian "Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi" berdasarkan dengan hasil rancangan yang sudah dibuat.

BAB 6 Pengujian dan Analisis

Pada bab ini membahas tentang teknik dan metode yang dilakukan dalam proses pengujian sistem yang sudah dibangun untuk memastikan sistem berfungsi sesuai dengan perancangan awal.

BAB 7 Penutup

Pada bab ini membahas tentang kesimpulan dan saran yang diperoleh dari rumusan masalah penelitian dengan melakukan analisis dan pengujian.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini menguraikan landasan kepustakaan dari penelitian yang akan dilakukan serta bahan penelitian sebelumnya yang didapat dari berbagai referensi. Kajian pustaka juga diuraikan dalam bab ini, yang merupakan rangkuman singkat dari berbagai referensi.

2.1 Tinjauan Pustaka

Kajian pustaka ini membahas tentang penelitian-penelitian yang sudah ada dan berkaitan dengan penelitian yang dilakukan penulis. Kajian pustaka yang dijadikan sebagai sumber adalah penelitian Nicholas Pantano pada tahun 2011 hingga 2012 yang berjudul "*Real Time Operating System On Arduino*". Journal yang dikaji berdasarkan syarat tumbuh stroberi adalah penelitian oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp pada tahun 2000 yang berjudul "Stroberi" beserta penelitian Sandhy Hermawan pada tahun 2016 yang berjudul "Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe". Berikut adalah perbedaan penelitian penulis dan pustaka yang diambil dijelaskan pada Tabel 2.1.

Tabel 2.1 Tinjauan Pustaka

NO	Nama Penulis, Tahun, dan Judul	Persamaan	Perbedaan	
			Penelitian Terdahulu	Rencana Penelitian
1.	Nicholas Pantano [2011-2012] Real Time Operating System On Arduino	Menerapkan Real Time Operating System pada Arduino	Meneliti utilitas RTOS pada arduino yang didemonstrasikan dengan menggunakan 2 sensor	Menerapkan RTOS pada sebuah sistem monitoring tanaman stroberi dengan menggunakan 3 sensor yaitu pencahayaan, suhu dan kelembaban tanah.
2.	Sandhy Hermawan[2015-2016] Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi	Menerapkan Tanaman Stroberi sebagai Objek Penelitian untuk Mendapatkan	Meneliti Perbandingan Stroberi dengan Ekstrak Jahe dan Konsentrasi Pektin terhadap Karakteristik	Menerapkan Batasan Nilai untuk Monitoring tanaman stroberi sesuai syarat tumbuh

	Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe	Syarat Tumbuh Baik Tanaman Stroberi	Minuman Fungsional Stroberi Jahe yang ihasilkan.	baik tanaman tersebut dan memberikan peringatan jika batasan nilai tersebut tidak terpernuhi

2.2 Dasar Teori

2.2.1 Stroberi

Tanaman stroberi merupakan tanaman buah berupa herba yang ditemukan pertama kali di Chili, Amerika. Salah satu spesies tanaman stroberi yaitu *Fragaria choiloensis L*. menyebar ke berbagai Negara Amerika, Eropa dan Asia. Selanjutnya spesies lain, yaitu *Fragaria vesca L*. lebih menyebar luas dibandingkan spesies lainnya. Jenis stroberi ini pula yang pertama kali masuk ke Indonesia. Stroberi yang penulis temukan di pasar swalayan adalah *hibrida* yang dihasilkan dari persilangan *Fragaria virgiana L*. var Duchesne asal Amerika Utara dengan *Fragaria Chiloensis L*. var Duchesne asal Chili. Persilangan itu menghasilkan *hybrid* yang merupakan stroberi modern (komersil) *Fragaria x annanassa var Duchesne* (Darwis, 2007).

Spesies tanaman stroberi yaitu *Fragaria chiloensis L.* menyebar ke berbagai Negara di Amerika, Eropa dan Asia. Sementara spesies lainnya yaitu *Fragaria vesca L* tersebar lebih luas dibandingkan spesies lainnya. Jenis stroberi *Fragaria vesca* yang pertama kali masuk di Indonesia (Budiman & Saraswati, 2010).

Stroberi adalah tanaman subtropis yang dapat beradaptasi dengan baik di dataran tinggi tropis yang memiliki temperature 17-20 derajat C dan disertai dengan curah hujan 600-700 mm/tahun. Stroberi juga membutuhkan kelembaban tanah yang baik untuk pertumbuhannya yang berkisar antara 80-90% dan lama penyinaran cahaya matahari yang dibutuhkan sekitar 8-10 jam setiap harinya (Hermawan, 2016).

Struktur akar tanaman stroberi terdiri atas pangkal akar (*collum*), batang akar (*corpus*), ujung akar (*apex*), bulu akar (*pilus radicalis*), serta tudung akar (*calyptra*). Tanaman stroberi berakar tunggang (*radix primaria*) terus tumbuh memanjang dan berukuran besar (Darwis, 2007).

Batang utama tanaman ini sangat pendek. Daun-daun terbentuk pada buku dan ketiak setiap daun terdapat pucuk *aksilar*. *Internode* sangat pendek sehingga jarak daun yang satu dengan yang lainnya sangat kecil dan memberi penampakan seperti rumpun tanpa batang. Batang utama dan daun yang tersusun rapat ini disebut *crown*. Ukuran *crown* berbeda-beda menurut umur, tingkat perkembangan tanaman, *kultivar* dan kondisi lingkungan pertumbuhan (Budiman & Saraswati, 2010).

Buah stroberi yang penulis kenal sebenarnya adalah buah semu, bukan buah yang sebenarnya. Buah stroberi yang dikenal masyarakat selama ini adalah *reseptakel* atau jaringan dasar bunga yang membesar. Buah yang sebenarnya adalah biji-biji kecil berwarna putih yang disebut dengan *achen*. *Achen* berasal dari sel kelamin betina yang telah diserbuki dan kemudian berkembang menjadi buah kerdil. *Achen* menempel pada permukaan reseptakel yang membesar (Setiani, 2007).

2.2.1.1 Syarat Tumbuh Stroberi

Sebelum menanam stroberi ada baiknya terlebih dahulu diketahui syarat-syarat tempat yang benar-benar sesuai bagi pertumbuhannya. Untuk itu diperlukan pengetahuan mengenai syarat lingkungan yang sesuai untuk tumbuh dan berproduksi. Sebab jika ternyata dikemudian hari ditemukan bahwa lingkungannya tidak cocok, maka segala jerih payah akan sia-sia. Misalnya, ia akan susah tumbuh dan merana. Ataupun mungkin saja walaupun tumbuh subur dengan daun yang cukup lebat, tetapi tanaman stroberi sedikit berbunga sampai tidak berbunga sama sekali. Atau dapat berbunga tapi bunga tersebut gagal menjadi buah (Hermawan, 2016).

Kondisi lingkungan tempat tanaman ini tumbuh dapat pula mempengaruhi rasa dan aroma buah stroberi, walaupun hal ini dipengaruhi oleh sifat *genetic* tanamannya. *Varietas* stroberi yang tumbuh di bawah cuaca cerah tetapi dingin pada malam harinya akan mempunyai rasa lebih enak di banding yang tumbuh di bawah udara berawan, lembab dan panas di malam hari (Soemadi, 1997).

2.2.1.2 Iklim

Tanaman stroberi dapat tumbuh dengan baik di daerah dengan curah hujan 600 – 700 mm/tahun. Lamanya penyinaran cahaya matahari yang di butuhkan dalam pertumbuhan adalah 8 – 9 jam setiap harinya. Stroberi adalah tanaman subtropis yang dapat beradaptasi dengan baik di dataran tinggi tropis yang memiliki temperatur 17 – 20˚C. Kelembaban udara yang baik untuk pertumbuhan tanaman stroberi anatara 80 – 90 % (Hermawan, 2016).

2.2.1.3 Kelembaban Tanah

Kelembaban yang berlebihan, berlangsung lama atau terjadi berulang kali, baik dalam bentuk hujan, embun atau kelembaban *relative* merupakan faktor yang sangat membantu perkembangan *epidemic* penyakit. Penyakit yang dipengaruhi kelembaban misalnya penyakit yang disebabkan oleh fungi (bercak daun, hawar, embung tepung, karat, antraknose), bakteri (bercak, hawar, busuk), dan nematode. Kelembaban mempengaruhi pertanaman tanaman inang menjadi sukulen dan rentan, meningkatkan sporulasi fungsi dan perbanyakan bakteri. Kelembaban rendah dalam beberapa hari, akan dapat mencegah terjadinya semua langkah-langkah perkembangan penyakit, sehingga epidemic terhambat atau terhenti.

2.2.1.4 Suhu

Kadang-kadang epidemi penyakit tanaman lebih berkembang karena pengaruh suhu yang lebih rendah atau lebih tinggi di banding dengan kisaran suhu optimum bagi tanaman inang. Kisaran suhu tertentu dapat menurunkan tingkat ketahan *horizontal* dan pada tingkat tertentu mungkin dapt menurunkan bahkan mematahkan ketahan *vertical* yang dibentuk oleh gen mayor. Tanaman yang tumbuh pada keadaan kisaran suhu tersebut akan mengalami 'stres' dan terdisposisi terhadap penyakit, sedangkan pathogen tumbuh dengan lebih baik dibanding inangnya. Suhu juga dapat menurunkan jumlah inokulum dan vector yang dapat bertahan hidup. Pengaruh suhu terhadap pathogen biasanya pada tingkat-tingkat yang berbeda dari pathogenesis, misalnya pada: perkecambahan spora atau penetasan telor, penetrasi ke inang, pertanaman, reproduksi, penyerangan inang atau pada sporulasi. Apabila pada tingkat kejadian, kisaran suhu menguntungkan, maka pathogen polisiklik dapat menyelesaikan daur penyakitnya dalam waktu pendek dan tetap member peluang berkembangnya epidemik

2.2.2 Konsep Dasar *System*

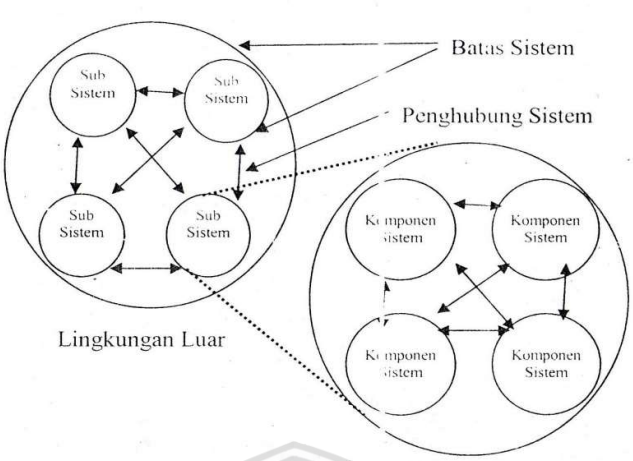
Suatu *system* adalah jaringan daripada elemen-elemen yang saling berhubungan, membentuk satu kesatuan yang untuk melaksanakan suatu tujuan pokok dari *system* tersebut (Mustakini, 2008).

System adalah sebuah tatanan yang terdiri atas sejumlah komponen fungsional (dengan tugas/fungsi khusus) yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses/pekerjaan tertentu (Kusrini, 2008).

Berdasarkan definisi-definisi tersebut dapat disimpulkan bahwa *system* merupakan jaringan daripada elemen-elemen yang terdiri atas sejumlah komponen fungsional (dengan tugas/fungsi khusus) yang saling berhubungan serta membentuk satu kesatuan dan melaksanakan suatu tujuan pokok secara bersama-sama sehingga dapat memenuhi suatu proses atau pekerjaan tertentu dari *system* tersebut.

2.2.2.1 Karakteristik *System*

Menurut Mustakini (2009:54), Suatu *system* mempunyai karakteristik. Karakteristik *system* adalah sebagai berikut ini:



Gambar 2.1 Karakteristik *System*

Sumber: (Mustakini, 2008)

1. Suatu *system* mempunyai komponen-komponen *system* (components) atau *subsystem-subsystem*.

   Suatu *system* terdiri dari sejumlah komponen-komponen yang saling berinteraksi, yang artinya saling bekerja sama dalam membentuk suatu kesatuan. Komponen *system* tersebut dapat berupa suatu bentuk *sub-system*.

2. Suatu *system* mempunyai batas *system* (boundary).

   Batasan *system* membatasi antara *system* yang satu dengan yang lainnya atau *system* dengan lingkungan luarnya.

3. Suatu *system* mempunyai lingkungan luar (environment).

   Lingkungan luar *system* adalah suatu bentuk apapun yang ada diluar ruang lingkup atau batasan *system* yang mempengaruhi operasi system tersebut.

4. Suatu *system* mempunyai penghubung (interface).

   Penghubung *system* merupakan media yang menghubungkan *system* dengan *sub-system* yang lain, dengan demikian dapat terjadi suatu integrasi *system* yang membentuk suatu kesatuan.

5. Suatu *system* mempunyai tujuan (goal).

   Suatu *system* pasti mempunyai tujuan (goals) atau sasaran *system* (objective). Sebuah *system* dikatakan berhasil apabila mengenai sasaran atau tujuannya, jika suatu *system* tidak mempunyai tujuan maka operasi *system* tidak akan ada gunanya.

2.2.2.2 Klasifikasi *System*

Menurut Mustakini (2009:53), Suatu *system* dapat diklasifikasikan:

1. Sistem abstrak (*abstact system*) dan sistem fisik *(phisical system*)

   Sistem abstrak adalah system yang berupa pemikiran atau ide-ide yang tidak tempak secara fisik, misalnya sistem teknologi yaitu sistem yang berupa pemikiran-pemikiran hubungan antara manusia dengan Tuhan. Sitem fisik merupakan system yang ada secara fisik.

2. Sistem Alami (*natural system*) dan Sistem Buatan Manusia *(human made system)*

   Sistem alami adalah sistem yang keberadaannya terjadi secara alami/natural tanpa campuran tangan manusia. Sedangkan sistem buatan manusia adalah sebagai hasil kerja manusia. Contoh sistem alamiah adalah sistem tata surya yang terdiri dari atas sekumpulan planet, gugus bintang dan lainnya. Contoh sistem abstrak dapat berupa sistem komponen yang ada sebagai hasil karya teknologi yang dikembangkan manusia.

3. Sistem pasti (*deterministic system*) dan sistem tidak tentu (*probobalistic system)*

   Sistem tertentu adalah sistem yang tingkah lakunya dapat ditentukan/diperkirakan sebelumnya. Sedangkan sistem tidak tentu sistem tingkah lakunya tidak dapat ditentukan sebelumnya. Sistem aplikasi komputer merupakan contoh sistem yang tingkah lakunya dapat ditentukan sebelumnya. Program aplikasi yang dirancangdan dikembangkan oleh manusia dengan menggunakan prosedur yang jelas, terstruktur dan baku.

4. Sistem Tertutup *(closed system*) dan Sistem Terbuka (*open system*)

   Sistem tertutup merupakan sistem yang tingkah lakunya tidak dipengaruhi oleh lingkungan luarnya. Sebaliknya, sistem terbuka mempunyai prilaku yang dipengaruhi oleh lingkungannya. Sistem aplikasi komputer merupakan sistem relative tertutup, karena tingkah laku sistem aplikasi komputer tidak dipengaruhi oleh kondisi yang terjadi diluar sistem.

2.2.3 *Monitoring*

Monitoring adalah penilaian secara terus menerus terhadap fungsi kegiatan-kegiatan program-program di dalam hal jadwal penggunaan input/masukan data oleh kelompok sasaran berkaitan dengan harapan-harapan yang telah direncanakan. Adapun pengertian *monitoring* menurut para ahli:

1. (Casely & Kumar, 1987) *Monitoring* merupakan program yang terintegrasi, bagian penting dipraktek manajemen yang baik dan arena itu merupakan bagian integral di manajemen sehari-hari.
2. (Clayton & Petry, 1983) *Monitoring* sebagai suatu proses mengukur, mencatat, mengumpulkan, memproses dan mengkomunikasikan

informasi untuk membantu pengambilan keputusan manajemen program/proyek.

3. (Oxfam, 1995) *Monitoring* adalah mekanisme yang sudah menyatu untuk memeriksa yang sudah untuk memeriksan bahwa semua berjalan untuk direncanakan dan memberi kesempatan agar penyesuaian dapat dilakukan secara metodologis.
4. (Kumorotomo, 2003) *Monitoring* adalah suatu proses pengumpulan dan menganalisis informasi dari penerapan suatu program termasuk mengecek secara reguler untuk melihat apakah kegiatan/program itu berjalan sesuai rencana sehingga masalah yang dilihat /ditemui dapat diatasi.

2.2.4 Pengenalan *Real Time Operating System* (RTOS)

Sistem operasi merupakan program komputer yang berfungsi memberikan layanan terhadap fungsi dasar komputer. Selain itu, sistem operasi juga memberi layanan bagi aplikasi atau program lain yang ada di komputer. Sistem operasi menghubungkan perangkat keras dengan perangkat lunak komputer yang berada di atasnya. Sistem operasi mempermudah pengguna memanfaatkan sumber daya komputer.

Embedded system menggunakan sistem operasi untuk mengatur eksekusi beberapa program dalam waktu bersamaan. Namun pada dasarnya prosessor pada sistem tertanam atau sistem komputer belum mampu melakukan eksekusi program secara bersama-sama. Sistem operasi akan berperan sebagai *scheduler* yang akan mengatur waktu eksekusi setiap program. Waktu eksekusi dibagi sedemikian rupa sehingga pergantian waktu antar program dapat diprediksi dengan tepat. Pada *embedded system* dikembangkan sistem operasi real time, dikenal dengan *Real Time Operating System* (Jatmiko, 2015).

2.2.5 *Real-Time Operating System* (RTOS)

RTOS merupakan perangkat lunak sistem yang berseluler mengatur sumber daya yang disediakan oleh perangkat keras dan menyediakan fasilitas pemrograman untuk digunakan oleh aplikasi. RTOS memiliki karakteristik yang berbeda dengan sistem operasi biasa, sehingga tidak semua sistem operasi bisa disebut sebagai sebuah sistem operasi waktu nyata (Laplante, 2008).

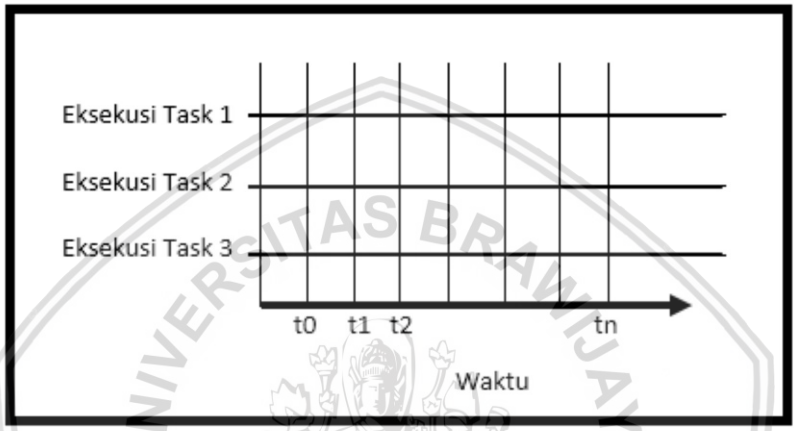
RTOS mengeksekusi program – program dalam sebuah pola yang teratur. RTOS bukan merupakan sistem operasi seperti pada umumnya yang terdapat pada komputer. Sistem operasi pada komputer akan bekerja sesaat ketika *power* masuk ke komputer, kemudian program berjalan. Lain halnya dengan RTOS, sistem operasi ini dijalankan oleh sebuah program otomatis. Program tersebut merupakan sebuah kernel. Ketika sistem dinyalakan, maka *kernel* akan menyala terlebih dahulu kemudian baru menyalakan *Real-Time Operating System*. Tugas utama dari RTOS ialah mengatur sumber daya yang ada meliputi processor, memori/register, serta hak akses menuju dua sumber daya tersebut. Selain itu, RTOS bertugas melakukan komunikasi dan sinkronisasi (Jatmiko, 2015).

2.2.6 Konsep Dasar *Real Time Operating System* (RTOS)

RTOS memiliki beberapa konsep dasar diantaranya *multitasking* dan scheduling. Berikut penjelasan lebih lanjut konsep-konsep dasar RTOS.

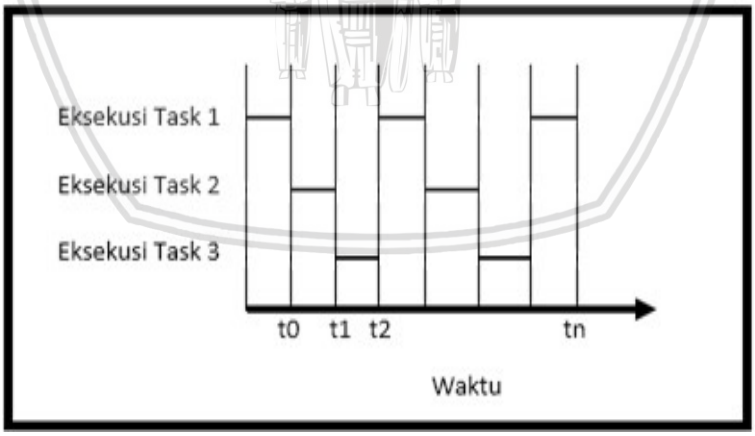
2.2.6.1 *Multitasking*

Sistem operasi memiliki komponen utama untuk mengatur adanya *multitasking*. *Multitasking* adalah kemampuan sistem operasi dalam menjalankan banyak *task* atau *thread* dalam rentan waktu tertentu. Konsep *multitasking* dan konkuren dapat dilihat pada Gambar 2.2 dan Gambar 2.3.



Gambar 2.2 Konsep *Multitasking*

Sumber: (diadaptasi dari Jatmiko, 2015)



Gambar 2.3 Konsep Konkuren

Sumber: (diadaptasi dari Jatmiko., 2015)

Pada Gambar 2.2 menunjukkan bagaimana konsep *multitasking* bekerja. Tiga *task* akan berjalan dalam waktu yang sama dan waktu eksekusi yang sama. Akan tetapi pada memori konvensional, konsep *multitasking* belum dapat dijalankan. Keterbatasan sumber daya prosesor dan memori membatasi

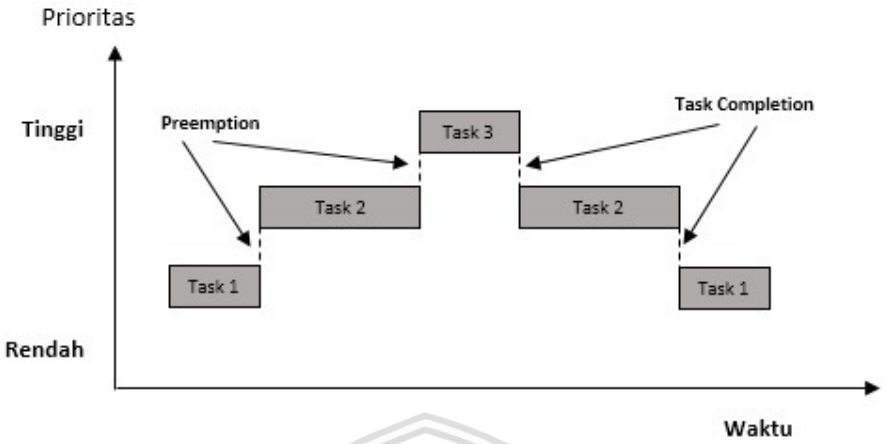
multitasking. Namun *task-task* yang ada secara bergantian akan dijalankan. Hal tersebut dinamakan dengan konkurensi. Konsep konkurensi digambarkan pada Gambar 2.3. Sistem operasi sudah mengetahui *task* mana saja yang akan dijalankan dalam rentan waktu tertentu. Selanjutnya sistem operasi membagi rentan waktu yang ada untuk mengeksekusi beberapa *task*. Sistem operasi mengatur kapan waktu eksekusi setiap *task* dan menyimpan posisi dari setiap *task* pada memori, sehingga ketika prosesor menjalankan suatu *task,* prosesor akan mengambil state terakhir dari *task* tersebut kemudian melanjutkan eksekusi *task* tersebut. Proses perpindahan antara eksekusi *task* dilakukan dengan cepat dan konkuren sehingga seolah - olah terlihat procesor melakukan beberapa *task* dalam rentan waktu bersamaan. *Embedded system* dapat menggunakan *RTOS* untuk menjalankan beberapa *task* secara konkuren. Pada akhirnya sistem tertanam seolah-olah dapat bekerja secara *multitasking.*

2.2.6.2 *Scheduling*

Scheduling merupakan konsep pembagian waktu eksekusi *task-task* yang ada. Untuk melakukan *scheduling* dibutuhkan adanya sebuah *scheduler*. *Scheduler* berfungsi untuk menentukan waktu eksekusi suatu *task,* lama eksekusi *task*, waktu suatu *task* ditunda (*suspend*), dan waktu *task* dijalankan kembali (*resume*). Pada penetuan waktu eksekusi *task* diperlukan suatu algoritma *scheduling*.

1. *Preemptive Priority-Based Scheduling*

   RTOS menggunakan algoritma *Preemptive Priority-Based Scheduling* sebagai algoritma default untuk *scheduling*. Algoritma ini mengatur pada setiap waktu *task* dengan *prioritas* paling tinggi akan dieksekusi. *Task* tersebut akan didahulukan dari *task* maupun yang juga sudah siap untuk dieksekusi. Gambar 2.4 menunjukkan bagaimana algoritma *Preemptive Priority-Based Scheduling* bekerja. *Task* dengan *prioritas* tinggi akan dieksekusi, sedangkan *task* lain akan ditunda dulu hingga *task* dengan *prioritas* lebih tinggi selesai dieksekusi. Pada RTOS, penentuan *prioritas* pada setiap *task* dilakukan pada saat pertama kali dibuat. Perubahan *prioritas* juga perlu memperhatikan pembagian *prioritas* yang sudah ada sehingga tidak menimbulkan *priority inversion*, *deadlock*, maupun kegagalan sistem.



Gambar 2.4 Algoritma *preemptive prioritt-based scheduling*

Sumber: (diadaptasi dari Jatmiko, 2015)

2.2.7 Objek Kernel *Real Time Operating System* (RTOS)

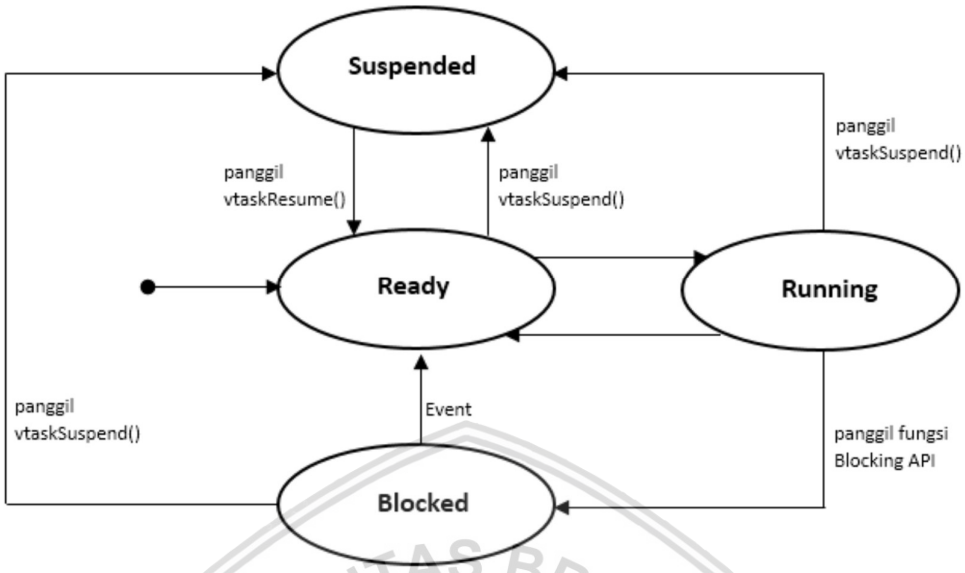
Kernel *object* merupakan blok-blok yang digunakan untuk membangun real time embedded system. Terdapat beberapa RTOS kernel *object* yaitu *task*, *Semaphore, message queue,* dan *co-routine* (Jatmiko, 2015). Berikut adalah penjelasan mengenai *task* dan *Semaphore.*

2.2.7.1 *Task*

Task merupakan *independent thread* dari sebuah proses yang dieksekusi. Setiap proses memiliki masing-masing *task*. Setiap *task* dapat berkompetisi dengan konkuren dengan *task* lain untuk menentukan waktu eksekusinya. *Task* dijalankan secara bergantian berdasarkan *prioritas* dan jadwal yang telah diatur oleh *scheduler* dengan menggunakan algoritma *scheduling. Scheduler* mengatur agar pada satu waktu hanya satu *task* yang dijalankan. Selanjutnya akan dijelaskan beberapa hal terkait suatu *task* diantaranya *task state* dan *task priorities*.

A. Task State

Setiap *task* berada pada suatu *state. State* tersebut akan menunjukkan status *task* saat ini. Terdapat empat status *task* dalam RTOS yaitu *ready, running, suspended*, dan *blocked.* Gambar 2.4 menunjukkan alur *state* yang dimiliki oleh setiap *task*.



Gambar 2.5 *Task State*

Sumber: (diadaptasi dari Jatmiko, 2015)

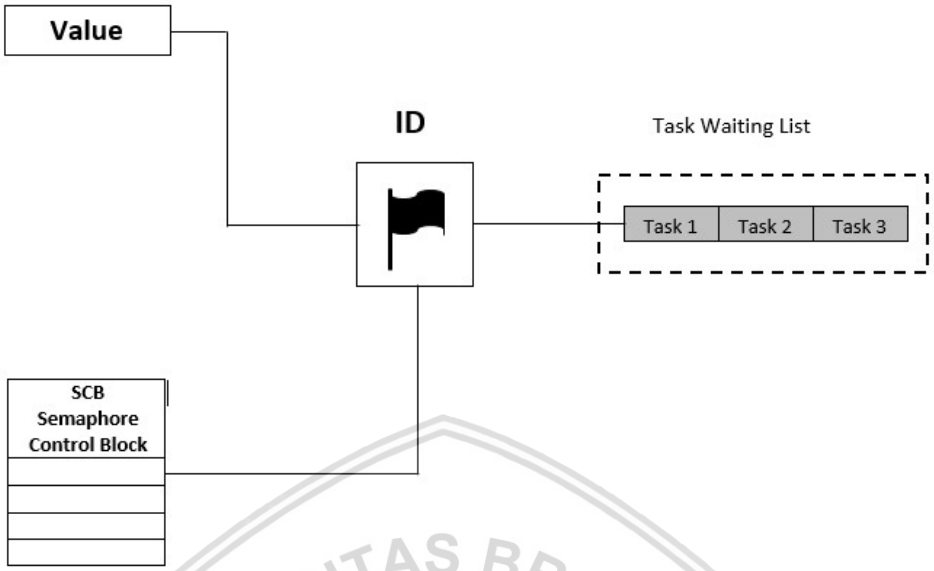
1) *Running State* : *Task* yang sedang berjalan/dieksekusi berada dalam *Running State.*
2) *Ready State* : Sesaat setelah *task* dibuat, *task* tersebut berada pada *ready state. Ready task* ialah *state* saat suatu *task* siap untuk dieksekusi.
3) *Blocked State* : Sebuah *task* berada pada *state* ini ketika *task* tersebut sedang menunggu *event* lain terjadi*. Event* yang terjadi meliputi temporal *event* maupun *external event*.
4) *Suspended* : *Suspended state* adalah *state* dimana *task* tersebut ditunda eksekusinya dalam waktu yang ditentukan. Penundaan ini terjadi ketika *task* menunggu *resource* untuk dikosongkan terlebih dahulu.

B. *Task* Priorities

Prioritas task menentukan urutan eksekusi *task-task* yang ada. Setiap *task* dapat memiliki *prioritas* dari 0 hingga nilai maximal priorities minus 1 (configMAX_PRIORITIES -1). Variabel configMAX_PRIORITIES – 1 didefinisikan dalam file FreeRTOSConfig.h. *Prioritas* terendah dimiliki oleh *idle task* yaitu *priority zero* sesuai dengan nilai pada variable tskIDLE_PRIORITY.

2.2.7.2 *Semaphore*

Semaphore adalah objek kernel dimana satu atau lebih *task* menggunakanya untuk sinkronisasi atar *task.* Setiap *Semaphore* memiliki nomer ID unik, sebuah *value binary* atau *count*, serta tersambung pada *waiting list* dan *Semaphore Control Block* (SCB). Gambar 2.5 menunjukkan ilustrasi dari *Semaphore*.



Gambar 2.6 *Semaphore*

Sumber: (diadaptasi dari Jatmiko, 2015)

Semaphore digunakan sebagai tanda bahwa *task* yang mengakusisinya memiliki hak untuk melaksanakan operasi tertentu juga mengakses *resource* yang ada. Akusisi *Semaphore* dibatasi oleh waktu dalam periode tertentu. *Semaphore* dapat diibaratkan sebagai kunci duplikat rumah yang dipegang oleh kepala keluarga. Anggota keluarga lain dapat meminjam kunci duplikat tersebut. Kunci duplikat tersedia dalam jumlah terbatas sehingga ketika sudah habis maka kepala keluarga tidak dapat memberikan lagi kuncinya. Pemakaian kunci juga terbatas, ketika sudah habis masa pakainya, maka anggota keluarga tidak bisa menjamin kuncinya sampai ada anggota lain mengembalikan kunci.

2.2.8 *Free Real Time Operating System* (FreeRTOS)

FreeRTOS adalah salah satu *Real Time Operating System* (RTOS) yang dapat dijalankan untuk *embedded system*. Sistem operasi ini dapat dipasang pada mikrokontroler. FreeRTOS menyediakan beberapa fitur utama suatu *Real Time Operating System*. Fitur tersebut diantaranya *real time scheduling functionality, inter-task communication, timing,* dan *synchronization primitives* (Jatmiko, 2015).

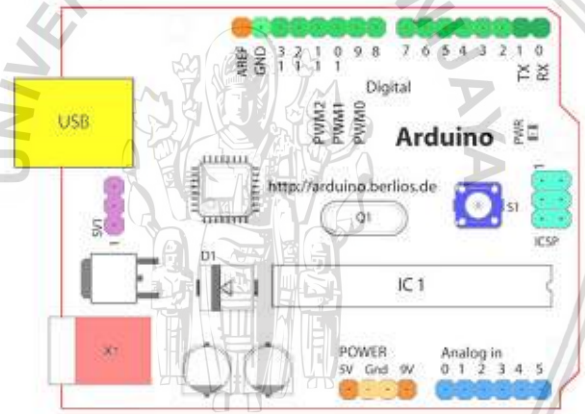
2.2.9 Arduino

Arduino merupakan *platform* yang terdiri dari *software* dan *hardware. Hardware* Arduino sama dengan *mikrocontroller* pada umumnya hanya pada arduino ditambahkan penamaan pin agar mudah diingat. Software Arduino merupakan *software open source* sehingga dapat di *download* secara gratis. *Software* ini digunakan untuk membuat dan memasukkan program ke dalam Arduino. Pemrograman Arduino tidak sebanyak tahapan *mikrocontroller* konvensional karena Arduino sudah didesain mudah untuk dipelajari, sehingga

para pemula dapat mulai belajar *mikrocontroller* dengan Arduino (Sulaiman, 2012).

2.2.10 *Hardware* Arduino

Arduino merupakan *platform open source* baik secara *hardware* dan *software*. Arduino terdiri dari mikrocontroller megaAVR seperti ATmega8, ATmega168, ATmega328, ATmega1280, dan ATmega 2560 dengan menggunakan Kristal osilator 16 MHz, namun ada beberapa tipe Arduino yang menggunakan Kristal osilator 8 MHz. Catu daya yang dibutuhkan untuk mensupply minimum system Arduino cukup dengan tegangan 5 VDC. Port arduino Atmega series terdiri dari 20 pin yang meliputi 14 pin I/O digital dengan 6 pin dapat berfungsi sebagai output PWM (Pulse Width Modulation) dan 6 pin I/O analog. Kelebihan Arduino adalah tidak membutuhkan *flash programmer external* karena di dalam chip *microcontroller* Arduino telah diisi dengan *bootloader* yang membuat proses *upload* menjadi lebih sederhana. Untuk koneksi terhadap komputer dapat menggunakan RS232 to TTL *Converter* atau menggunakan Chip USB ke Serial converter seperti FTDI FT232 (Sulaiman, 2012).

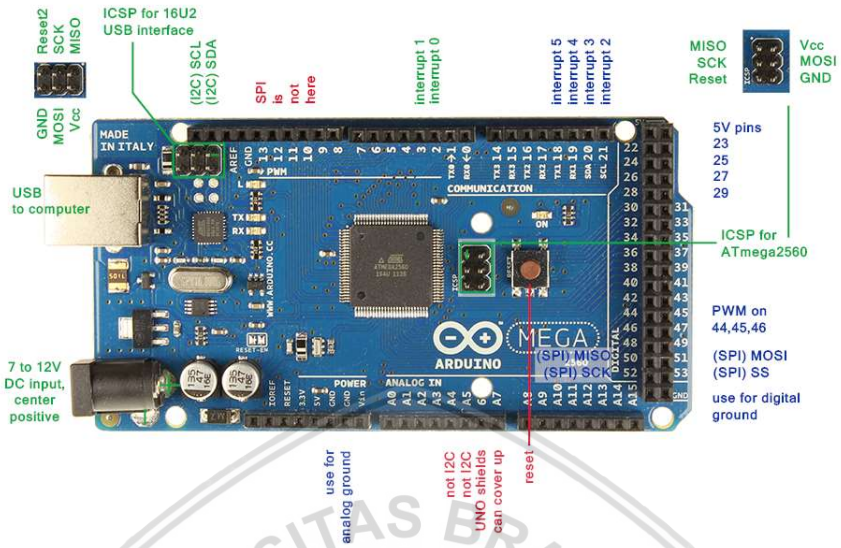


Gambar 2.7 *Hardware* Arduino

Sumber: Djuandi (2011:5)

Arduino board sendiri telah tersedia dalam banyak jenis baik yang sudah berkoneksi USB maupun serial. Contoh Arduino yang terkoneksi dengan USB seperti: Arduino Mega, Arduino Uno, Arduino Duemilanove, Arduino Diecimila, Arduino NG Rev. C , Arduino FIO, dan Arduino lilypad. Untuk lilypad memiliki ukuran sebesar kancing baju dan anti air sehingga dapat dicuci. Sedangkan Arduino Severino merupakan contoh untuk yang terkoneksi secara serial.

2.2.11 Arduino Mega



Gambar 2.8 Arduino Mega

Sumber: (https://forum.arduino.cc)

Arduino Mega 2560 adalah papan pengembangan mikrokontroller yang berbasis Arduino dengan menggunakan chip ATmega2560. Board ini memiliki pin I/O yang cukup banyak, sejumlah 54 buah digital I/O pin (15 pin diantaranya adalah PWM), 16 pin analog input, 4 pin UART (serial port hardware). Arduino Mega 2560 dilengkapi dengan sebuah oscillator 16 Mhz, sebuah port USB, power jack DC, ICSP header, dan tombol reset. Board ini sudah sangat lengkap, sudah memiliki segala sesuatu yang dibutuhkan untuk sebuah mikrokontroller. Dengan penggunaan yang cukup sederhana, anda tinggal menghubungkan *power* dari USB ke PC anda atau melalui adaptor AC/DC ke jack DC.

Chip ATmega2560 pada Arduino Mega 2560 Revisi 3 memiliki memori 256 KB, dengan 8 KB dari memori tersebut telah digunakan untuk *bootloader.* Jumlah SRAM 8 KB, dan EEPROM 4 KB, yang dapat di baca-tulis dengan menggunakan EEPROM library saat melakukan pemrograman. Arduino Mega 2560 memiliki jumlah pin terbanyak dari semua papan pengembangan Arduino. Mega 2560 memiliki 54 buah digital pin yang dapat digunakan sebagai input atau output, dengan menggunakan fungsi pinMode(), digitalWrite(), dan digital(Read). Pin-pin tersebut bekerja pada tegangan 5V, dan setiap pin dapat menyediakan atau menerima arus sebesar 20mA, dan memiliki tahanan pull-up sekitar 20-50k ohm (secara *default* dalam posisi disconnect). Nilai maximum adalah 40mA, yang sebisa mungkin dihindari untuk menghindari kerusakan chip mikrokontroller.

Beberapa pin memiliki fungsi khusus:

1. **Serial**, Komunikasi antara Arduino Uno dan Komputer dapat dilakukan melalui port USB. Dalam hal ini, Arduino Uno tidak hanya bisa mengolah data dari pin I/O secara independ. Arduino mega memiliki 4 serial yang

masing-masing terdiri dari 2 pin. Serial 0: pin 0 (RX) dan pin 1 (TX). Serial 1: pin 19 (RX) dan pin 18 (TX). Serial 2: pin 17 (RX) dan pin 16 (TX). Serial 3: pin 15 (RX) dan pin 14 (TX). RX digunakan untuk menerima dan TX untuk transmit data serial TTL. Pin 0 dan pin 1 adalah pin yang digunakan oleh chip USB-to-TTL ATmega16U2.

2. **External Interrups**, Suatu permintaan khusus pada mikrokontroller untuk melakukan sesuatu, jika terjadi interupsi maka program akan menghentikan dahulu apa yang sedang dikerjakan dan melakukan apa yang diminta oleh yang menginterupsi. External interrups memiliki beberapa pin yaitu pin 2 (untuk interrupt 0), pin 3 (interrupt 1), pin 18 (interrupt 5), pin 19 (interrupt 4), pin 20 (interrupt 3), dan pin 21 (interrupt 2). Dengan demikian Arduino Mega 2560 memiliki jumlah interrupt yang cukup melimpah: 6 buah. Gunakan fungsi attachInterrupt() untuk mengatur interrupt tersebut.
3. **PWM**, Suatu metode untuk mendapatkan bentuk sinyal analog dari sinyal digital. Fungsi ini dapat membuat transisi antar state hidup dan mati menjadi lebih halus. Pin dari PWM adalah Pin 2 hingga 13 dan 44 hingga 46, yang menyediakan output PWM 8-bit dengan menggunakan fungsi analogWrite()
4. **SPI**, Sebuah antarmuka bus yang biasa digunakan untuk mengirim data antara mikrokontroler dan perangkat kecil seperti register geser , sensor , dan kartu SD. Pin dari SPI adalah Pin 50 (MISO), 51 (MOSI), 52 (SCK), dan 53 (SS) mendukung komunikasi SPI dengan menggunakan SPI Library
5. **LED**, Pin 13. Pada pin 13 terhubung built-in led yang dikendalikan oleh digital pin no 13. Set HIGH untuk menyalakan led, LOW untuk memadamkan nya.

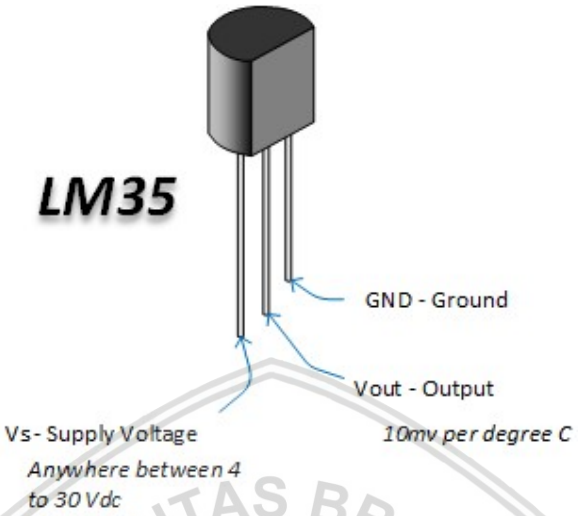
Arduino Mega 2560 R3 memiliki 16 buah input analog. Masing-masing pin analog tersebut memiliki resolusi 10 bits (jadi bisa memiliki 1024 nilai). Secara default, pin-pin tersebut diukur dari ground ke 5V, namun bisa juga menggunakan pin AREF dengan menggunakan fungsi analogReference(). Beberapa in lainnya pada board ini adalah:

1. **AREF**. Sebagai referensi tegangan untuk input analog.
2. **Reset**. Hubungkan ke LOW untuk melakukan reset terhadap mikrokontroller. Sama dengan penggunaan tombol reset yang tersedia.

2.2.12 Sensor Suhu LM35

Sensor suhu LM35 adalah komponen elektronika yang memiliki fungsi untuk mengubah besaran suhu menjadi besaran listrik dalam bentuk tegangan. Sensor LM35 memiliki keakuratan tinggi dalam pembacaan suhu dan kemudahan dalam perancangan jika dibandingkan dengan sensor-sensor suhu yang lain, Sensor LM35 juga mempunyai keluaran berupa impedansi yang rendah dan

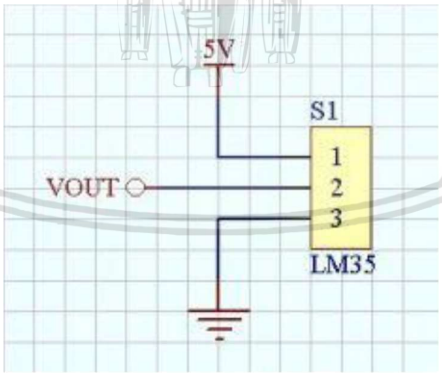
linieritas yang tinggi sehingga dapat dengan mudah dihubungkan dengan rangkaian kendali khusus serta tidak memerlukan penyetelan lanjutan.



Gambar 2.9 Bentuk fisik sensor LM35

Sumber: (probots.co.in)

Pada Gambar 2.9 adalah bentuk fisik dari LM35. 3 pin LM35 menujukan fungsi masing-masing pin diantaranya, pin 1 berfungsi sebagai sumber tegangan kerja dari LM35, pin 2 atau tengah digunakan sebagai tegangan keluaran atau Vout dengan jangkauan kerja dari 0 Volt sampai dengan 1,5 Volt dengan tegangan operasi sensor LM35 yang dapat digunakan antar 4 Volt sampai 30 Volt. Keluaran sensor ini akan naik sebesar 10 mV setiap derajad celcius sehingga diperoleh persamaan VLM35 = Suhu* 10 mV.



Gambar 2.10 Skematik Rangkaian LM35

Sumber: **(probots.co.in)**

Pada Gambar 2.10 adalah rangkaian sederhana pada sensor LM35. Vout adalah tegangan keluaran sensor yang terskala linear terhadap suhu terukur, yakni 10 milivolt per 1 derajad celcius. Jadi jika Vout = 530mV, maka suhu terukur adalah 53 derajad Celcius. Dan jika Vout = 320mV, maka suhu terukur adalah 32 derajad Celcius. Tegangan keluaran ini bisa langsung diumpankan sebagai masukan ke rangkaian pengkondisi sinyal seperti rangkaian penguat operasional dan rangkaian

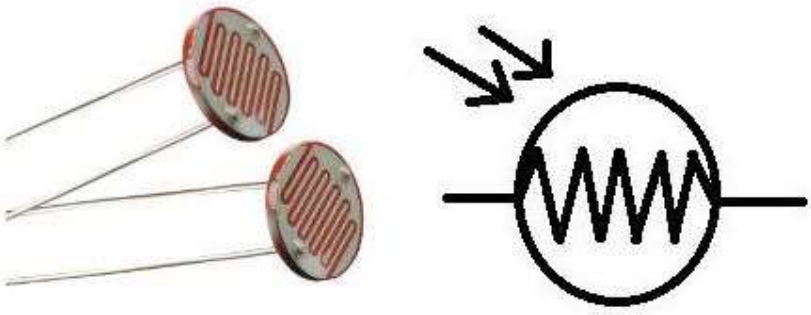
filter, atau rangkaian lain seperti rangkaian pembanding tegangan dan rangkaian Analog-to-Digital Converter.

Rangkaian dasar tersebut cukup untuk sekedar bereksperimen atau untuk aplikasi yang tidak memerlukan akurasi pengukuran yang sempurna. Akan tetapi tidak untuk aplikasi yang sesungguhnya. Terbukti dari eksperimen yang telah penulis lakukan, tegangan keluaran sensor belumlah stabil. Pada kondisi suhu yang relatif sama, jika tegangan suplai penulis rubah (naikkan atau turunkan), maka Vout juga ikut berubah. Memang secara logika hal ini sepertinya benar, tapi untuk instrumentasi hal ini tidaklah diperkenankan. Dibandingkan dengan tingkat kepresisian, maka tingkat akurasi alat ukur lebih utama karena alat ukur seyogyanya dapat dijadikan patokan bagi penggunanya. Jika nilainya berubah-ubah untuk kondisi yang relatif tidak ada perubahan, maka alat ukur yang demikian ini tidak dapat digunakan.

2.2.13 Sensor Cahaya LDR

Light Dependent Resistor (LDR) ialah jenis resistor yang berubah hambatannya karena pengaruh cahaya. Besarnya nilai hambatan pada sensor cahaya LDR tergantung pada besar kecilnya cahaya yang diterima oleh LDR itu sendiri. Bila cahaya gelap nilai tahanannya semakin besar, sedangkan cahayanya terang nilainya menjadi semakin kecil. LDR adalah jenis resistor yang biasa digunakan sebagai detektor cahaya atau pengukur besaran konversi cahaya. LDR terdiri dari sebuah cakram semikonduktor yang mempunyai dua buah elekrtroda pada permukaannya.

Resistansi LDR berubah seiring dengan perubahan intensitas cahaya yang mengenainya. Dalam keadaan gelap resistansi LDR sekitar 10 MΩ dan dalam keadaan terang sebesar 1KΩ atau kurang. LDR terbuat dari bahan semikonduktor seperti senyawa kimia *cadmium sulfide*. Dengan bahan ini energi dari cahaya yang jatuh menyebabkan lebih banyak muatan yang dilepas atau arus listrik meningkat, artinya resistansi bahan telah mengalami penurunan. Seperti halnya resistor konvensional, pemasangan LDR dalam suatu rangkaian sama persis seperti pemasangan resistor biasa. Simbol LDR dapat dilihat seperti gambar berikut:



Gambar 2.11 Bentuk Fisik Dan Simbol LDR

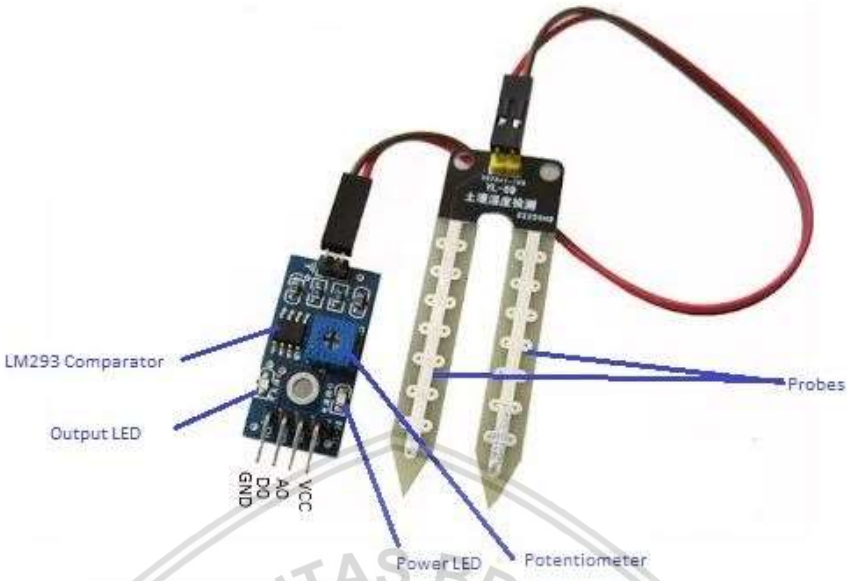
Sumber: (data sheet CDS Light-Dependent Photoresistors, 2010)

Salah satu komponen yang menggunakan sensor adalah LDR (*Light Dependent Resistor*), adalah suatu komponen elektronika yang memiliki hambatan yang dapat berubah sesuai perubahan intensitas cahaya, resistensi dari LDR akan menurun jika ada penambahan intensitas cahaya yang mengenainya. Pada dasarnya komponen ini merupakan suatu resistor yang memiliki nilai hambatan bergantung pada jumlah cahaya yang jatuh pada permukaan sensor tersebut. LDR dapat dibuat dari semikonduktor beresistensi tinggi yang tidak dilindungi dari cahaya. Jika cahaya yang mengenainya memiliki frekuensi yang cukup tinggi, foton yang diserap oleh semikonduktor akan menyebabkan elektron memiliki energi yang cukup untuk meloncat ke pita konduksi. Elektron bebas yang dihasilkan dan pasangan lubangnya akan mengalirkan listrik, sehingga menurunkan resistansinya.

Komponen yang menggunakan sensor cahaya berikutnya adalah *Photo Transistor*, secara sederhana adalah sebuah transistor bipolar yang memakai kontak (*junction*) *base-collector* yang menjadi permukaan agar dapat menerima cahaya sehingga dapat digunakan menjadi konduktivitas transistor. Secara lebih detail *Photo Transistor* merupakan sebuah benda padat pendeteksi cahaya yang memiliki gain internal. Hal ini yang membuat foto transistor memiliki sensivitas yang lebih tinggi dibandingkan *photodiode*, dalam ukuran yang sama. Alat ini dapat menghasilkan sinyal analog maupun sinyal digital. Photo Transistor sejenis dengan transistor pada umumnya, bedanya pada Photo Transistor dipasang sebuah lensa pemfokus sinar pada kaki basis untuk memfokuskan sinar jatuh pada pertemuan PN (komponenelektronika.biz, 2012).

2.2.14 Soil Moisture Sensor SEN0114

Soil Moisture Sensor adalah sensor yang dapat mendeteksi kelembaban tanah disekitarnya. Sensor ini terdiri dari dua probe untuk melewatkan arus listrik dalam tanah, kemudian membaca resistansinya untuk mendapatkan nilai tingkat kelembaban. Semakin banyak air membuat tanah lebih mudah menghantarkan listrik (resistansi kecil), sedangkan tanah yang kering sangat sulit menghantarkan listrik (resistansi besar).



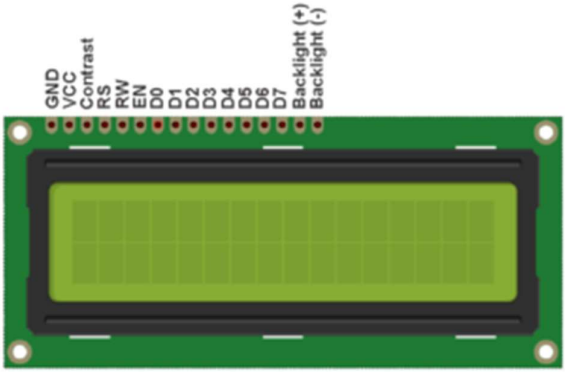
Gambar 2.12 Soil Moisture Sensor

Sumber: (probots.co.id)

Sensor ini sangat membantu untuk mengingatkan tingkat kelembaban di pabrik atau untuk memantau kelembaban tanah untuk pertanian. IO Expansion Shield adalah perisai untuk menghubungkan sensor dengan Arduino (DFrobot, 2012). Pada sensor kelembaban memiliki 3 macam kondisi keluaran untuk dapat menemukan nilai pada satuan% RH, yaitu kering = 0 ~ 358, lembab = 359 ~ 460 dan basah = 461 ~ 495 seperti yang ditunjukkan pada spesifikasi berikut ini:

1. Power supply: 3.3v or 5v
2. Output voltage signal: 0~4.2v
3. Current: 35mA
4. Pin definition:
    a) Analog output (Blue wire).
    b) GND (Black wire).
    c) Power (Red wire).
5. Size: 60x20x5mm.
6. Value range:
    a) 0~358 : dry soil
    b) 359~460 : humid soil
    c) 461~495 : in water

2.2.15 LCD 16X2



Gambar 2.13 LCD 16X2

Sumber: http://Circuits4you.com/

LCD (Liquid Crystal Display) adalah suatu jenis media tampil yang menggunakan kristal cair sebagai penampil utama. LCD sudah digunakan diberbagai bidang misalnya alal–alat elektronik seperti televisi, kalkulator, atau pun layar komputer. Pada postingan aplikasi LCD yang digunakan ialah LCD dot matrik dengan jumlah karakter 2 x 16. LCD sangat berfungsi sebagai penampil yang nantinya akan digunakan untuk menampilkan status kerja alat.

Adapun fitur yang disajikan dalam LCD ini adalah :

1. Terdiri dari 16 karakter dan 2 baris.
2. Mempunyai 192 karakter tersimpan.
3. Terdapat karakter generator terprogram.
4. Dapat dialamati dengan mode 4-bit dan 8-bit.
5. Dilengkapi dengan back light.

2.2.16 Modul I2C

Gambar 2.14 Modul I2C

Sumber: http://www.14core.com

Inter Integrated Circuit atau sering disebut I^2C adalah standar komunikasi serial dua arah menggunakan dua saluran yang didisain khusus untuk mengirim maupun menerima data. Sistem I^2C terdiri dari saluran SCL (*Serial Clock*) dan SDA (*Serial Data*) yang membawa informasi data antara I^2C dengan pengontrolnya. Piranti yang dihubungkan dengan sistem I2C Bus dapat dioperasikan sebagai *Master* dan *Slave*. *Master* adalah piranti yang memulai *transfer* data pada I^2C Bus dengan membentuk sinyal *Start*, mengakhiri *transfer* data dengan membentuk sinyal *Stop*, dan membangkitkan sinyal *clock*. *Slave* adalah piranti yang dialamati *master.*

Sinyal *Start* merupakan sinyal untuk memulai semua perintah, didefinisikan sebagai perubahan tegangan SDA dari "1" menjadi "0" pada saat SCL "1". Sinyal *Stop* merupakan sinyal untuk mengakhiri semua perintah, didefinisikan sebagai perubahan tegangan SDA dari "0" menjadi "1" pada saat SCL "1".

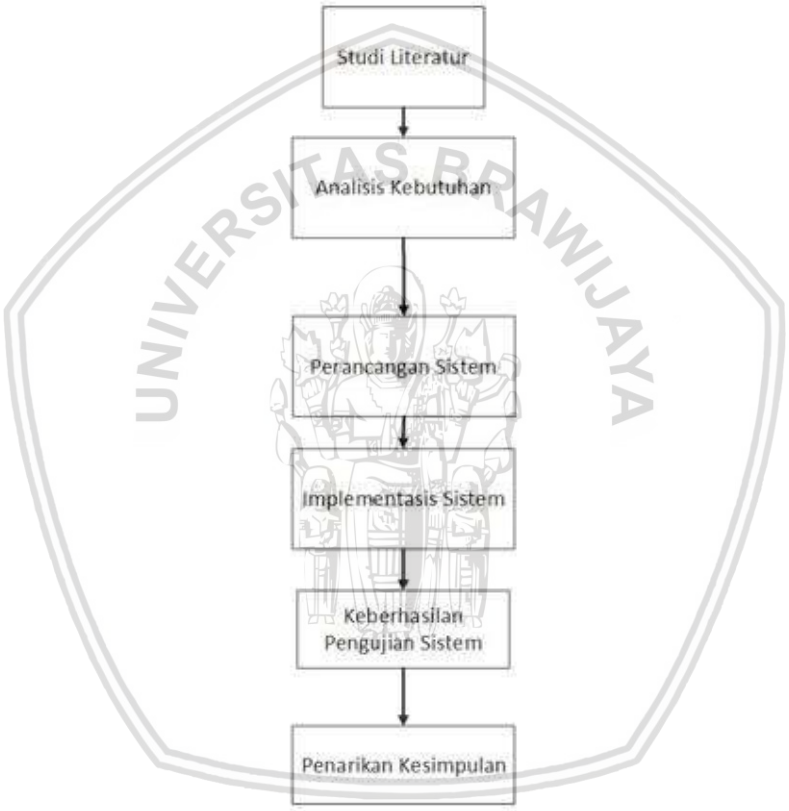
Sinyal dasar yang lain dalam I^2C Bus adalah sinyal *acknowledge* yang disimbolkan dengan ACK Setelah transfer data oleh *master* berhasil diterima *slave, slave* akan menjawabnya dengan mengirim sinyal *acknowledge*, yaitu dengan membuat SDA menjadi "0" selama siklus *clock* ke 9. Ini menunjukkan bahwa *Slave* telah menerima 8 bit data dari *Master*.

Dalam melakukan *transfer* data pada I^2C Bus, penulis harus mengikuti tata cara yang telah ditetapkan yaitu:

1. *Transfer* data hanya dapat dilakukan ketikan Bus tidak dalam keadaan sibuk.
2. Selama proses transfer data, keadaan data pada SDA harus stabil selama SCL dalam keadan tinggi. Keadaan perubahan "1" atau "0" pada SDA hanya dapat dilakukan selama SCL dalam keadaan rendah. Jika terjadi perubahan keadaan SDA pada saat SCL dalam keadaan tinggi, maka perubahan itu dianggap sebagai sinyal *Start* atau sinyal *Stop*

BAB 3 METODOLOGI

Di dalam bab ini membahas langkah-langkah dan prosedur yang digunakan untuk menyelesaikan peneitian "Implementasi *System Real Time* untuk *Monitoring* Pencahayaan, Suhu, dan Kelembaban pada Tanaman Stroberi". Dalam bab ini juga berfungsi untuk menyelesaikan permasalahan yang timbul selama proses penelitian dengan cara pengumpulan data. Di dalam metodologi penelitian yang digunakan penulis terdiri dari enam bagian yaitu studi literatur, analisis kebutuhan, perancangan, implementasi, pengujian analisis dan pengambilan kesimpulan.



Gambar 3.1 Diagram Alir Metodologi Penelitian

3.1 Studi Literatur

Studi literatur adalah tahap untuk mencari referensi dari penelitian-penelitian sebelumnya sebagai pendukung penelitian yang akan dilakukan. Mempelajari dan memahami literatur yang berhubungan dengan penelitian yang dibuat, seperti pembacaan sensor LDR, sensor LM35, sensor SEN0114, penggunaan Arduino Mega, proses penjadwalan RTOS. Literatur dapat dipelajari melalui jurnal, ebook, makalah, buku, situs, tugas akhir, dan lain-lain.

3.2 Rekayasa Kebutuhan

Rekayasa kebutuhan merupakan proses analisis pada kebutuhan yang diperlukan sistem pada penelitian. Pada tugas akhir kebutuhan yang diperlukan yaitu meliputi kebutuhan fungsional sistem, komponen perangkat lunak (*software)* maupun perangkat keras *(hardware*). Komponen *software* dan *hardware* yang digunakan berfungsi sebagai konfigurasi dari sistem yang dirancang oleh penuis.

3.2.1 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras yang diperlukan untuk membangun sistem monitoring tanaman stroberi adalah sebagai berikut:

1. Laptop yang telah terinstall software diatas
2. Arduino Mega
3. Project Board
4. Sensor Cahaya LDR
5. Sensor Kelembaban Tanah SEN0114
6. Sensor Suhu LM35
7. LCD
8. LED RGB

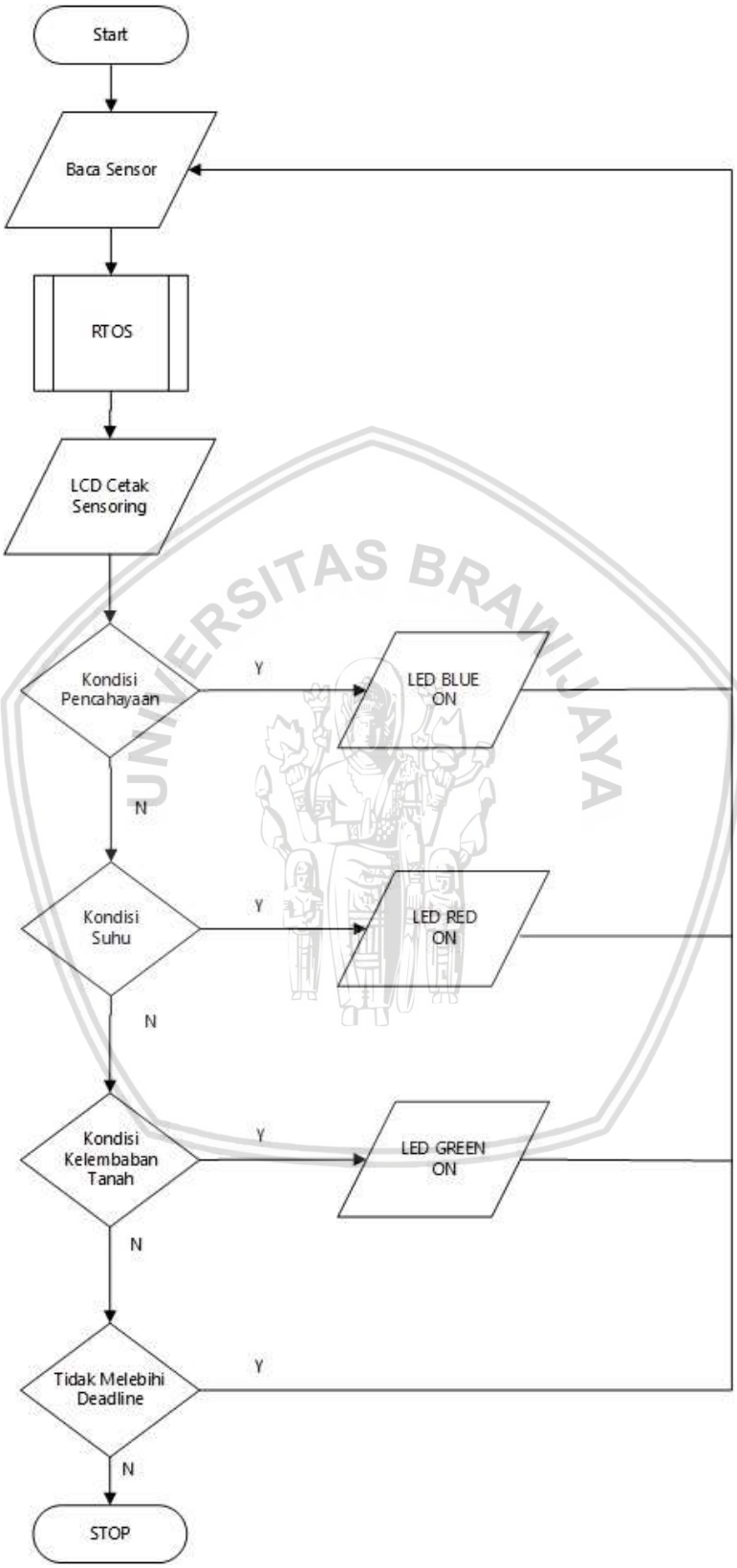
3.2.2 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak yang diperlukan untuk membangun sistem monitoring tanaman stroberi adalah sebagai berikut:

1. Sistem Operasi Windows 10
2. Arduino IDE
3. Arduino_FreeRTOS.h
4. Semaphr.h
5. LiquidCrystal_I2C.h.

3.3 Langkah-langkah Perancangan Sistem

Perangkat sistem merupakan tahap yang dilakukan setelah analisis kebutuhan. Perencanaan sistem dilakukan apabila seluruh kebutuhan sistem telah terpenuhi. Tujuan dilakukan perencanaan sistem agar mempermudah pada saat tahap iplementasi karena dilakukan secara sistematis dan terstruktur. Berikut ini merupakan gambar yang menunjukkan diagram aliran dari sistem kerja dan rancangan sistem yang akan dibuat.



Gambar 3.2 Diagram Alir Sistem Kerja

Pada perancangan sistem ini pengolahan data diproses menggunakan Arduino Mega. Data berupa pencahyaan, suhu dan kelembaban tanah pada tanaman stroberi. Data pencahayaan, data suhu dan data kelembaban tanah didapat dari sensor LDR sensor LM35 dan sensor SEN0114 saat arduino sudah aktif. Kedua data tersebut diproses dengan menggunakan *task* pada RTOS, dimana sudah ada pada program mikrokontroler Arduino Mega.

RTOS dalam sistem akan memproses data input pencahayaan suhu dan kelembaban tanah pada tanaman stroberi berdasarkan kondisi yang sudah ditentukan dalam sistem. Hasil pengolahan data tersebut berupa akuisisi data sensoring pencahayaan, suhu dan kelembaban tanah yang di tampilkan melalui LCD dan ditandai dengan nyala lampu LED.

Real-Time Operating System (RTOS) dalam sistem akan mengatur penjadwalan tiap *task.* Tiap *task* akan diberi *prioritas* yang berbeda agar sistem dapat menjalankan *task* dimana tidak akan ada data yang bertabrakan.

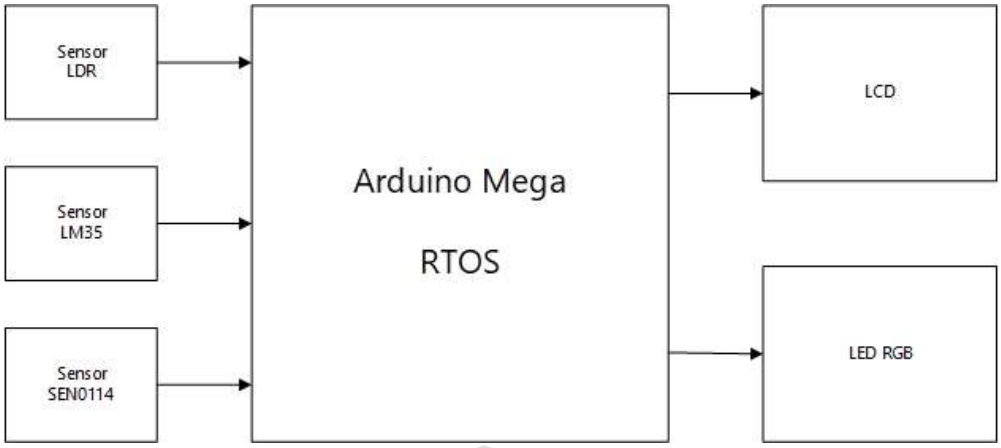
Hasil pengolahan data oleh sistem akan digunakan untuk menentukan kondisi pencahayaan, suhu dan kelembaban tanah pada tanaman stroberi. LCD digunakan untuk menampilkan akuisisi data dari 3 sensor. Apabila kondisi cahaya tidak sesuai dengan kondisi yang sudah di tentukan sistem, maka lampu LED berwarna biru menyala, apabila kondisi suhu tidak sesuai dengan kondisi yang ditentukan sistem, maka lampu LED berwarna merah akan menyala, lalu kelembaban tanah tidak sesuai dengan kondisi yang ditentukan sistem, maka lampu LED berwarna hijau akan menyala.

3.4 Langkah-langkah Implementasi Sistem

Implementasi dilakukan berdasarkan peracangan sistem yang dijelaskan pada Gambar 3.1 langkah yang harus dilakukan yaitu:

3.4.1 Konfigurasi Perangkat Keras

Pada tahap konfigurasi perangat keras, yang harus dilakukan adalah dengan merancang sensor LDR, sensor LM35 dan sensor SEN0114 untuk disambungkan ke perangkat Arduino mega. Sensor-sensor akan disambungkan melalui media kabel jumper male to female dan male to male. Penyambungan dilakukan berdasarkan datasheet masing-masing sensor dan Arduino mega. Blok diagram perancangan perangkat keras dapat dilihat pada Gambar 3.3.



Gambar 3.3 Blok Diagram Perancangan Perangkat Keras

3.4.2 Konfigurasi Perangkat Lunak

Pada tahap konfigurasi perangkat lunak, yang harus dilakukan adalah penginstallan software Arduino IDE yang dapat diunduh melalui website arduino. Setelah itu, dilanjutkan pengkodingan program dengan menggunakan bahasa C. untuk mengkonfigurasi jalannya sistem monitoring tanaman stroberi, mengimplementasikan Real-time operating system untuk mengatur scheduling *task* dan menentukan kondisi untuk output berdasarkan perhitungannya.

3.5 Langkah-Langkah Pengujian dan Analisis

Pengujian akan dilakukan atas beberapa tahapan mulai dari pengujian akusisi data pada sensor cahaya, suhu dan kelembaban tanah, pengujian RTOS dan pengujian perbandingan waktu eksekusi sistem monitoring yang menggunakan RTOS dengan sistem monitoring tanpa RTOS.

3.6 Penarikan Kesimpulan

Kesimpulan diperoleh ketika seluruh pengujian telah selesai dilakukan. Isi kesimpulan didasarkan pada rumusan masalah pada penelitian yang dilakukan.

BAB 4 REKAYASA KEBUTUHAN

Bab ini menjelaskan secara detail mengenai kebutuhan-kebutuhan yang wajib terpenuhi untuk perancangan dan implementasi sistem.

4.1 Gambaran Umum Sistem

Sistem yang di bangun merupakan *prototype* implementasi sistem *real time* untuk *monitoring* tanaman stroberi berdasarkan intensitas cahaya, suhu dan kelembaban tanah dengan memanfaatkan *real time operating system* mengunakan library FreeRtos.h. Sistem ini di bangun menggunakan beberapa sensor untuk mengakuisisi data sesuai kondisi pada tanaman stroberi. Hasil yang di peroleh, di proses oleh mikrokontroler kemudian di tampilkan dalam LCD dan di tandai dengan lampu peringatan LED RGB.

4.1.1 Perspektif Sistem

Sistem berhasil apabila sistem dapat mengakuisisi data cahaya, suhu dan kelemababan tanah tanaman stroberi, mengatur penjadwalan tiap *task* berdasarkan *prioritas*, agar dapat di eksekusi secara konkuren menggunakan *Real Time Operating System*. Selain itu, sistem dapat memberikan peringatan jika pencahayaan, suhu dan kelembaban tanah tidak sesuai dengan kondisi yang ditentukan sistem.

4.1.2 Ruang Lingkup

Ruang lingkup dari sistem adalah penerapan *Real Time Operating System* (RTOS) sebagai *scheduler* untuk mengatur tugas tiap *task*, urutan *task* berdasarkan *prioritas* dan *delay task* yang akan di eksekusi pada Arduino mega, mengontrol LED RGB sebagai peringatan, dan sistem akan menampilkan data sensoring pada LCD.

4.1.3 Karakteristik Pengguna

Sistem ini berupa prototipe dan pengujian dilakukan penulis selaku pembuat sistem. Sistem ini merupakan sistem interferensi manusia, dimana user hanya melihat hasil dari sistem.

4.1.4 Lingkungan Operasi Sistem

Untuk mendapatkan kondisi lingkungan yang mendukung maka, sistem ini dijalankan didalam pot tanaman stroberi supaya didapatkan hasil pembacaan sensor berdasarkan lingkungan pada ruang terbuka.

4.1.5 Batasan Perancangan dan Implementasi

Batasan perancangan dan implementasi pada sistem yang buat meliputi:

1. Mikrokontroller yang digunakan adalah Arduino Mega

2. Sistem menggunakan modul sensor LDR untuk mendeteksi paparan cahaya, sensor LM35 untuk mendeteksi suhu lingkungan dan sensor SEN0114 untuk mendeteksi kelembaban tanah.
3. Sistem menggunakan FreeRTOS yang bertugas mengatur jadwal antar *task* agar dapat berjalan secara konkuren.
4. Output dari sistem berupa sensoring paparan cahaya, suhu, dan kelembaban tanah yang akan di tampilkan dalam LCD dan nyala lampu LED RGB yang diseuaikan dengan kondisi.

4.2 Rekayasa Kebutuhan

Merupakan kebutuhan yang wajib dimiliki sebuah sistem agar sistem dapat bekerja sesuai dengan tujuan. Pada sub bab ini menjelaskan tentang kebutuhan fungsional dari sistem, perangkat keras, dan perangkat lunak.

4.2.1 Kebutuhan Fungsional Sistem

Kebutuhan fungsional sistem merupakan kebutuhan yang harus dimiliki suatu sistem dalam memberikan suatu layanan yang seharusnya. Kebutuhan fungsional yang wajib dipenuhi adalah sebagai berikut.

1. Sistem dapat melakukan proses akusisi data dari paparan cahaya, suhu pada lingkungan dan kelembaban pada tanah pada tanaman stroberi.

   Fungsi ini berguna sebagai data input dari sensor LDR, LM35 dan Soil Moisture. Dimana sensor LDR digunakan untuk mendeteksi paparan cahaya, sensor LM35 digunakan untuk mendeteksi suhu lingkungan dalam satuan *celcisus* dan sensor SEN0114 digunakan untuk mendeteksi tingkat kelembaban pada tanah.

2. Sistem dapat menjadwalkan tiap *task* dengan RTOS.

   Fungsi ini digunakan untuk melakukan penjadwalan pada *task* – *task* yang terdapat dalam sistem. Penjadwalan yang dilakukan berdasarkan *prioritas* dan *delay task* yang diberikan pada *task. Task* yang memiliki *prioritas* terbesar akan dieksekusi terlebih dahulu dan dilanjutkan oleh *task* dengan *prioritas* terbesar berikutnya.

3. Sistem dapat menampilkan output pada LCD dan LED RGB

   Fungsi ini berguna untuk menampilkan *output* dari sistem melalui LCD dan LED RGB sehingga lebih mudah dimengerti oleh pengguna. LED bertugas sebagai peringatan dimana apabila kondisi pencahayaan, suhu dan kelembaban tanah tidak sesuai dengan kondisi yang telah di tentukan sistem sedangkan, LCD digunakan sebagai penampil akuisisi data pada sensor LDR, sensor LM35, dan sensor SEN0114.

4.2.2 Kebutuhan Perangkat Keras

Kebutuhan perangkat keras pada sistem ini meliputi kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional adalah kebutuhan yang wajib terpenuhi agar sistem dapat bekerja dan berfungsi dengan baik, jika kebutuhan fungsional tidak terpenuhi maka sistem tidak akan berjalan. Kebutuhan non-fungsional merupakan kebutuhan yang tidak wajib terpenuhi oleh sistem, karena apabila tidak ada kebutuhan ini maka sistem tetap bisa berjalan. Kebutuhan fungsional dan non-fungsional dapat dilihat pada Tabel 4.1 dan Tabel 4.2.

Tabel 4.1 Kebutuhan fungsional perangat keras

Kebutuhan Fungsional	Fungsi
Arduino Mega	• Dapat menerima input dari sensor cahaya, suhu dan kelembaban untuk menentukan output sistem. • Menjadwalkan tiap *task* dengan *Real Time Operating System*. Fungsi ini mengharuskan sistem dapat membuat *task* berjalan dengan konkuren berdasarkan priority yang telah di*setting*. Seperti *task* a dikerjakan terlebih dahulu, setelah selesai *task* akan dikerjakan dan seterusnya.
Sensor LDR	• Membaca paparan cahaya dengan satuan volt.
Sensor LM35	• Membaca suhu dengan satuan celcius.
Sensor SEN0114	• Membaca tingkat kelembaban tanah pada tanaman stroberi dengan satuan %.
LCD	• Menampilkan sensoring dengan satuan fahrenheit, kelembaban tanah dan pencahayaan
LED RGB	• Menyalakan lampu berwarna biru jika paparan cahaya tidak sesuai dengan kondisi yang sudah ditentukan • Menyalakan lampu berwarna merah jika suhu tidak sesuai dengan kondisi yang sudah ditentukan • Menyalakan lampu berwarna hijau jika kelembaban tanah tidak sesuai dengan kondisi yang sudah ditentukan
Tegangan	• Memberikan sumber tegangan yang sesuai untuk perangkat keras yang digunakan.

Tabel 4.2 Kebutuhan Non-fungsional perangkat keras

Kebutuhan Non-Fungsional	Fungsi
Laptop	• Menampilkan hasil output dari sistem melalui serial monitor pada Arduino IDE, sehingga data output dari sistem dapat dianalisis. • Sebagai sumber tegangan pada mikrokontroller arduino mega.

4.2.3 Kebutuhan Perangkat Lunak

Kebutuhan perangkat lunak meliputi kebutuhan non-fungsional sistem. Berikut adalah beberapa kebutuhan non-fungsional sistem ditampilkan pada Tabel 4.3.

Tabel 4.3 Kebutuhan Non-fungsional perangkat lunak

Kebutuhan	Fungsi
Arduino_FreeRTOS.h	• Membuat dan mengatur penjadwalan *task* • Menentukan *prioritas* pada *task* • Menentukan delay pada *task*
Semaphr.h	• *Check & create Semaphore* • *Take & give Semaphore*

BAB 5 PERANCANGAN DAN IMPLEMENTASI

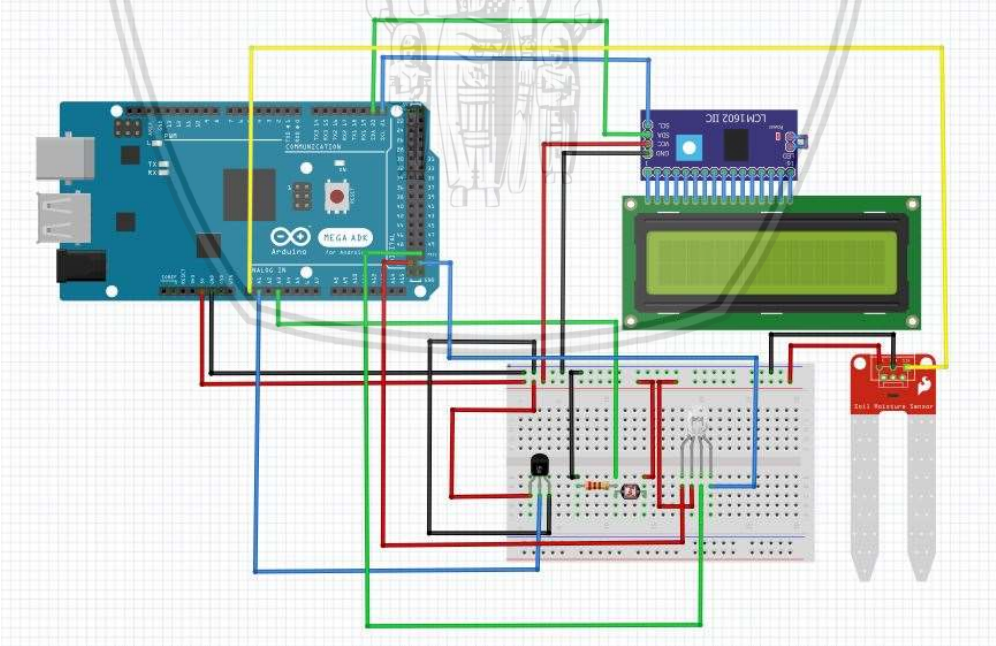
Bab ini akan membahas mengenai perancangan dari sistem yang akan dibuat dan berdasarkan hasil perancangan tersebut akan dilakukan implementasi sistem yang terdiri dari implementasi perangkat lunak dan perangkat keras.

5.1 Perancangan Sistem

Perangkat yang dibangun secara umum adalah sebuah alat yang dapat memantau tanaman stroberi berdasarkan pencahyaan, suhu dan kelembaban tanah secara *real time* dan akurat. Alat ini di lengkapi dengan LED sebagai penanda kondisi yang di deteksi oleh tenaman stroberi dan LCD sebagai penampil hasil sensoring. Sistem pada alat ini dilengkapi dengan RTOS sebagai *scheduler* tiap *task* pada sistem. Proses perancangan ini dibagi menjadi 2 tahap, yaitu Perancangan perangkat keras dan perancangan perangkat lunak.

5.1.1 Perancangan Perangkat Keras

Perancangan sistem ini memerlukan suatu infrastruktur sebagai penunjang yang saling berkaitan satu dengan yang lain yaitu perangkat keras / *Hardware*. Perancangan perangkat keras bertujuan untuk akuisi data pada pencahayaan, suhu dan kelembaban tanah yang di implementasikan dengan *Real Time Operating System*. Skema perancangan perangkat keras secara keseluruhan dapat dlihat pada Gambar 5.1 Berikut



Gambar 5.1 Skema perancangan perangkat keras

Pada gambar tersebut seluruh komponen digabung menjadi satu sistem yang saling bekerja sama untuk mencapai tujuan yang sama. Perangkat keras pada sistem ini antara lain, Arduino mega, sensor LM35, sensor LDR, sensor SEN1004, LED RGB, dan LCD. Arduino Mega bertugas sebagai pusat pengkontrol kerja perangkat lain dan pusat pemprosesan data pada sistem. Koneksi pin pada Gambar 5.1 diatas ditunjukan pada Tabel 5.1 berikut.

Tabel 5.1 Koneksi pin perancangan perangkat keras

Pin Arduino Mega	Pin LM35	Pin LDR	Pin SEN0114	Pin LED RGB	Pin LCD I2C
Vcc	Vcc	Vcc	Vcc	Vcc	Vcc
GND	GND	GND	GND		GND
A0			Vout		
A1	Vout				
A3		Vout			
51				LED Green	
52				LED Red	
53				LED Blue	
SCL					SCL
SDA					SDA

Untuk desain prototipe perangkat keras dibuat wadah berbahan acrylic, dimana sensor LDR, sensor LM35, sensor SEN0114 dan LED RGB berada di dalam dan diberi celah agar sensor tembus ke luar, sementara sensor SEN0114 diberi kabel panjang, agar sensor dapat tertancap pada pot tanaman stroberi, peletakan tersebut agar sensor dapat mendeteksi pencahyaan, suhu dan kelembaban tanah yang ada didepannya. lalu LED RGB diberi celah kecil berbentuk bulat untuk memperjelas nyala lampu yang di tampilkan. Karena bahan acrylic yang tembus pandang maka, LCD juga diletakan di dalam untuk mempermudah melihat output yang di tampilkan.

5.1.2 Perancangan Perangkat Lunak

Perangkat lunak yang terdapat pada sistem ini berupa program pada mikrokontroler Arduino Mega yang menerapkan RTOS sebagai *scheduler* tiap *task* agar memperoleh hasil yang presisi.

Perancangan perangkat lunak dimulai dari input cahaya yang di dapat dari sensor LDR, suhu yang di dapat dari sensor LM35 dan kelembaban tanah yang di dapat dari sensor SEN0114. Data dari input ini akan di simpan pada suatu variabel. Kemudian data tersebut akan di implementasikan pada *Real-Time Operating System* (RTOS) yang telah ditentukan kondisi untuk memutuskan peringatan pada tanaman stroberi berdasarkan pencahyaan, suhu dan kelembaban tanah.

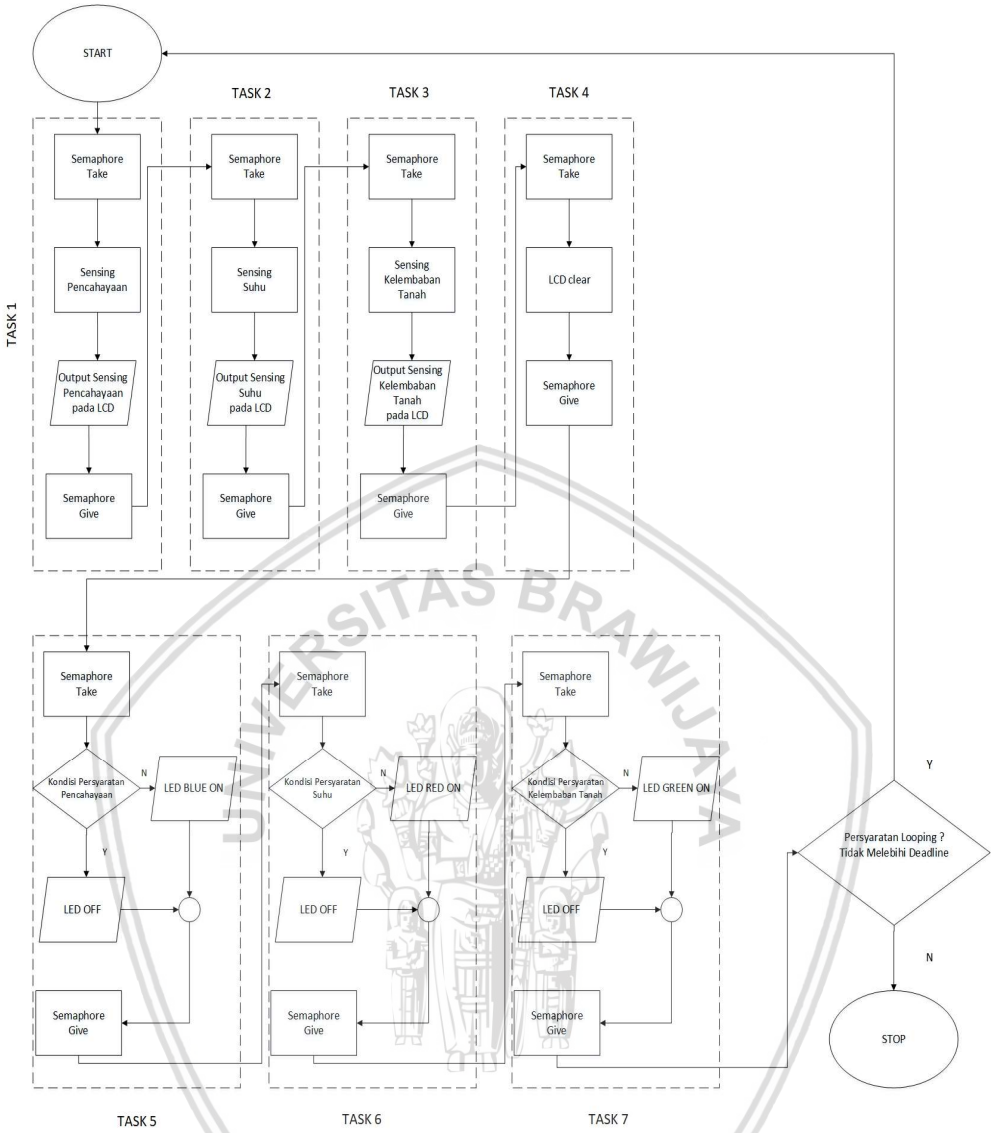
Untuk pengimplementasian peringatan dalam sistem, nantinya akan dibuat kondisi berdasarkan syarat tumbuh stroberi yang digunakan untuk memproses hasil sensoring dari cahaya, suhu dan kelembaban tanah.

Untuk pengimplementasian *Real-Time Operating System* (RTOS), sistem akan dibagi menjadi 7 *task*, dimana : *task* pertama bertugas melakukan sensing pencahyaan pada tanaman stroberi dan menampilkan hasil sensing cahaya pada layar LCD, *task* kedua, bertugas melakukan sensing suhu di sekitar tanaman stroberi dan menampilkan sensing suhu pada layar LCD, *task* ketiga bertugas melakukan sensing pada kelembaban tanah tanaman stroberi dan menampilkan sensing kelembaban tanah pada layar LCD, *task* keempat bertugas untuk menghapus layar lcd, *task* kelima bertugas untuk memberi peringatan jika cahaya tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *task* pertama, *task* keenam bertugas untuk memberi peringatan jika suhu tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *task* kedua, *task* ketujuh bertugas untuk memberi peringatan jika kelembaban tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *task* ketiga. Ketujuh *task* ini nantinya akan diberi *prioritas* berbeda antar *task* pembacaan sensor dan *task* lampu peringatan. Selain pembagian *task* dan *prioritas*, akan diimplementasikan *Semaphore* yang menjaga *task* yang berjalan secara konkuren untuk tidak saling mengganggu ketika sedang mengakses sumber daya yang sama.

Output dari sistem ini berupa hasil sensing dari 3 sensor yakni LDR, LM35 dan SEN0114 yang di tampilkan pada LCD. Kemudian peringatan jika pencahyaan, suhu dan kelembaban tanah tidak sesuai dengan syarat tumbuh tanaman stroberi yang di implementasikan pada LED RGB. Lampu menyala pada saat kondisi cahaya, suhu dan kelembaban tanah tidak sesuai dengan syarat tumbuh stroberi. Nyala lampu biru untuk pencahyaan, nyala lampu merah untuk suhu dan nyala lampu hijau untuk kelembaban tanah. Jika ketiga unsur tersebut memenuhi untuk persyaratan tumbuh tanaman stroberi, maka lampu LED RGB mati.

5.1.2.1 Perancangan *Real Time Operating System*

Pada proses perancangan RTOS terdiri dari beberapa tahapan yaitu menentukan berapa *task* yang akan dibuat, menentukan tugas tiap *task*, mentukan *priority* tiap *task*, dan mengatur *delay* tiap *task*nya. *Prioritas* dengan nilai terbesar akan dieksekusi terlebih dahulu. Diagram alir perancangan RTOS dapat dilihat pada Gambar 5.2.



Gambar 5.2 Diagram alir perancangan *Real Time Operating System*

Seperti Gambar 5.2, sistem akan dibagi menjadi 7 *task* dimana tiap *task* memiliki tugasnya masing-masing tiap *task*nya menggunakan sumber yang sama yaitu *Semaphore*. Dengan penggunaan *Semaphore* tiap *task* akan memakai sumber daya secara bergantian agar dapat berjalan secara konkuren. Ketika sistem dijalankan, *task* dengan *prioritas* terbesar akan melakukan pengambilan *Semaphore* dan mulai mengeksekusi *task* hingga selesai. Setelah eksekusi *task* tersebut selesai, *task* akan melepaskan *Semaphore* untuk digunakan oleh *task* dengan *prioritas* terbesar nomer 2 dan seterusnya. Pada persyaratan *looping* sistem akan kembali melakukan perulangan jika semua *task* berjalan tanpa melebihi *deadline* yang telah ditentukan tetapi apabila *task* yang di eksekusi melebihi *deadline* yang telah ditentukan maka sistem tidak akan berjalan dan mengalami kegagalan sistem. Untuk pembagian tugas, *prioritas* serta *deadline*nya lebih jelasnya dapat dilihat pada Tabel 5.2.

Tabel 5.2 Pembagian tugas, *prioritas* dan *deadline* pada sistem

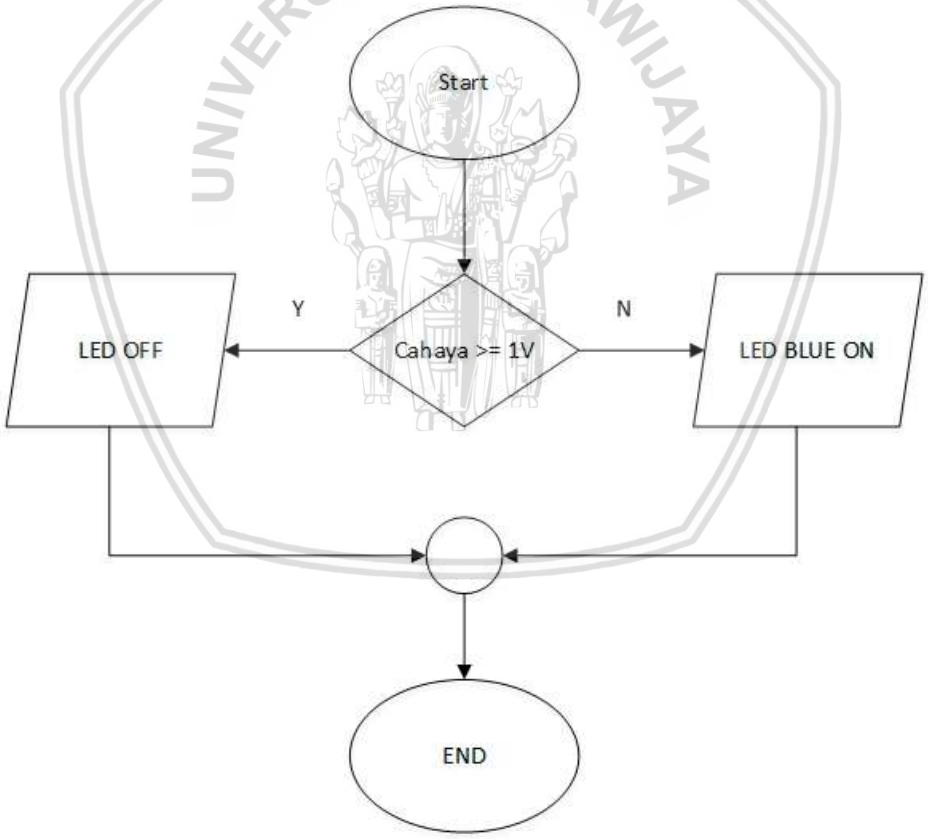
No	*Task*	Tugas	*Prioritas*	*Deadline*
1	*Task*LDR	Bertugas melakukan sensing pencahayaan pada tanaman stroberi dan menampilkan hasil sensing suhu pada layar LCD	3	1000ms
2	*Task*LM35	Bertugas melakukan sensing suhu pada tanaman stroberi dan menampilkan hasil sensing suhu pada layar LCD	3	
3	*Task*Mois	Bertugas melakukan sensing kelembaban tanah pada tanaman stroberi dan menampilkan hasil sensing suhu pada layar LCD	3	
4	LCD	Bertugas untuk menghapus layar LCD	3	2000ms
5	*Tracehole*LDR	Bertugas untuk memberi peringatan jika pencahyaan tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *Task*LDR	2	1000ms
6	*Tracehole*LM35	Bertugas untuk memberi peringatan jika suhu tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *Task*LM35	2	
7	*Tracehole*Mois	Bertugas untuk memberi peringatan jika kelembaban tanah tidak sesuai dengan syarat tumbuh tanaman stroberi yang didapatkan pada *Task*Mois	2	

Pada Tabel 5.2, tugas sensing dari tiap sensor yaitu *Task*LDR *Task*LM35 dan *Task*Mois diberi *prioritas* terbesar sebab sistem membutuhkan data sensoring untuk bisa mengerjakan tugas berikutnya yaitu tugas memberikan peringatan berupa lampu LED GRB dan menentukan *deadline* untuk semua sensing data sebesar 1000ms. Pada *Task*LCD diberi *prioritas* yang sama dengan tugas sensing data tetapi diberikan *deadline* lebih besar yaitu 2000ms tujuanya adalah agar data sensing pada *Task*LDR *Task*LM35 dan *Task*Mois dapat terus di perbaharui setiap 2 detik, jika pada *Task*LCD diberikan *deadline* yang sama yakni 1000ms maka sebelum *Task*LDR *Task*LM35 dan *Task*Mois menampilkan nilai pada LCD, nilai akan langsung terhapus dan mata manusia tidak bisa melihat hasil sensing karena

terlalu cepatnya perpindahan antar tiap *task* pada sistem. Kemudian yang terakhir adalah tugas lampu peringatan yaitu *Tracehole*LDR *Tracehole*LM35 dan *Tracehole*Mois yang di eksekusi setelah mendapatkan akuisisi data dari *Task*LDR *Task*LM35 dan *Task*Mois. Sistem akan melalukan perulangan jika semua *Task* tidak melewati *deadline* yang telah ditentukan.

5.1.2.2 Perancangan Lampu Peringatan pada LED RGB

Pada proses perancangan lampu peringatan pada LED RGB peneliti memakai journal yang dikaji berdasarkan syarat tumbuh stroberi oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp dan journal yang berjudul "Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe" oleh Sandhy Hermawan untuk menentukan batasan pencahyaan, suhu, dan kelembaban tanah pada tanaman stroberi. Flowchart perancangan lampu peringatan pada LED RGB dapat dilihat pada Gambar 5.3 untuk kondisi suhu, Gambar 5.4 untuk kondisi pencahyaan dan Gambar 5.5 untuk kondisi kelembaban tanah.

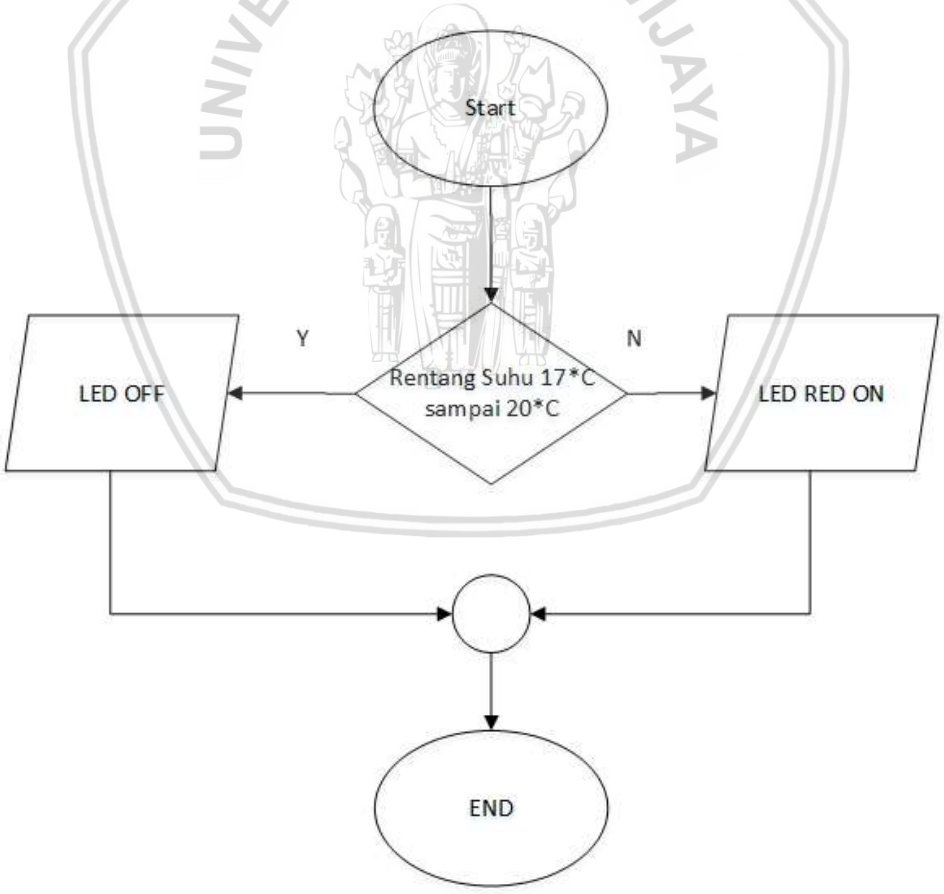


Gambar 5.3 Flowchart batasan nilai untuk pencahayaan

Pada proses perancangan lampu peringatan untuk LDR, peneliti memberi batasan nilai sensing kurang dari 1 V, jika sensing pada pencahyaan sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing pencahayaan tidak

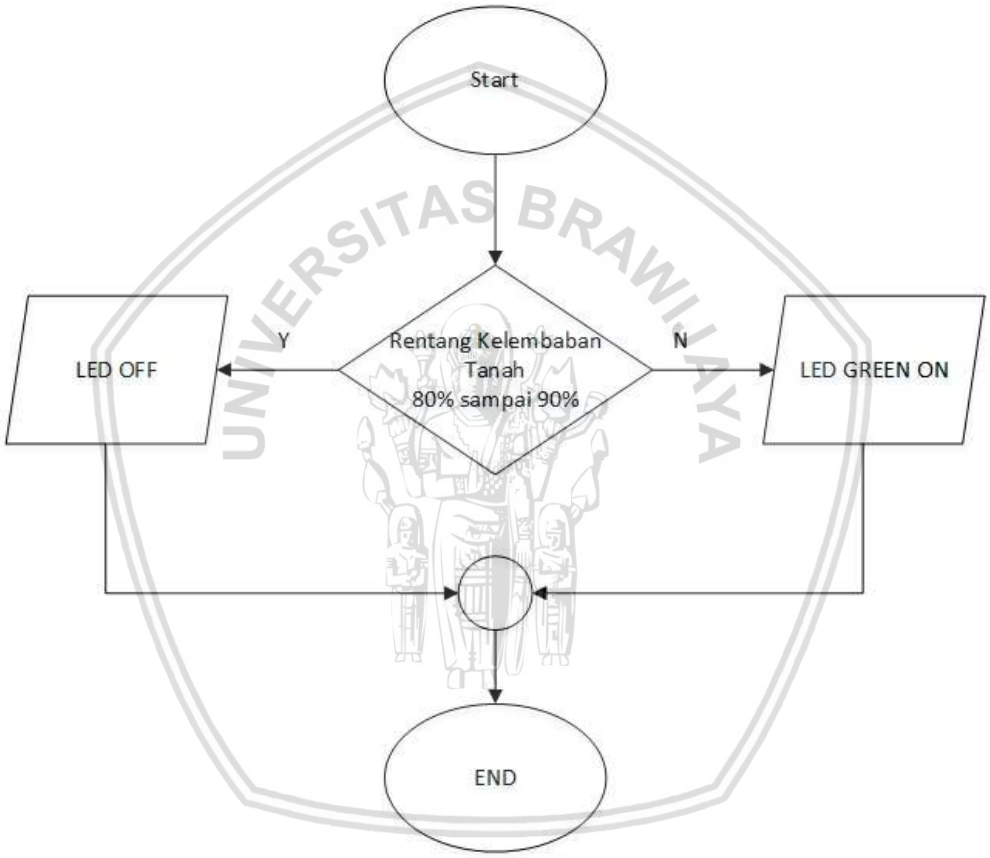
sesuai dengan batasan nilai tersebut maka LED BLUE akan menyala itu berarti pencahyaan yang didapat oleh tanaman stroberi tidak terpenuhi. LDR sensor sebetulnya adalah sebuah Resistor (hambatan) yang nilai resistansi nya tergantung pada cahaya yang jatuh pada permukaannya. Makin besar intensitas cahaya yang masuk (makin terang), maka nilai hambatannya makin kecil. Oleh karena itu, peneliti pasangkan LDR dengan suatu resistor (10 kilo ohms) membentuk suatu voltage divider (pembagi tegangan), maka peneliti bisa mengukur tegangan pada titik yang ditandai tersebut sebagai suatu input sensor cahaya. Nilai tegangan ini akan tergantung pada nilai hambatan dari LDR, yang mana hal ini tergantung pada cahaya yang masuk.

Jika cahaya yang masuk sangat terang sekali, nilai hambatan LDR akan menjadi sangat kecil sekali, sehingga tegangan pada titik ukur mendekati 5V. Sedangkan pada keadaan sangat gelap, hambatan LDR menjadi sangat besar sekali sehingga tegangan di titik ukur mendekati 0V. Dalam kenyataannya peneliti mendapatkan titik tegangan di antara 0V dan 5V sesuai intensitas cahaya yang masuk. Sensor cahaya ini dapat peneliti gunakan untuk pemantauan terhapad pencahyaan sinar matahari pada tanaman stroberi.



Gambar 5.4 Flowchart batasan nilai untuk suhu

Pada proses perancangan lampu peringatan untuk LM35, peneliti memberi batasan nilai antara 17 *C sampai dengan 20 *C, jika sensing pada suhu sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing suhu tidak sesuai dengan batasan nilai tersebut maka LED RED akan menyala itu berarti suhu yang didapat oleh tanaman stroberi tidak sesuai untuk syarat tumbuh baik tanaman stroberi. Batasan nilai tersebut di dapat dari journal yang di kaji oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp dan journal yang berjudul "Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe" oleh Sandhy Hermawan.



Gambar 5.5 Flowchart batasan nilai untuk kelembaban tanah

Pada proses perancangan lampu peringatan untuk kelembaban tanah, peneliti memberi batasan nilai antara 80 % sampai dengan 90 %, jika sensing pada kelembaban tanah sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing kelembaban tanah tidak sesuai dengan batasan nilai tersebut maka LED GREEN akan menyala itu berarti kelembaban tanah yang didapat oleh tanaman stroberi tidak sesuai untuk syarat tumbuh baik tanaman stroberi. Batasan nilai tersebut di dapat dari journal yang di kaji oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp dan journal yang berjudul "Kajian Perbandingan Stroberi

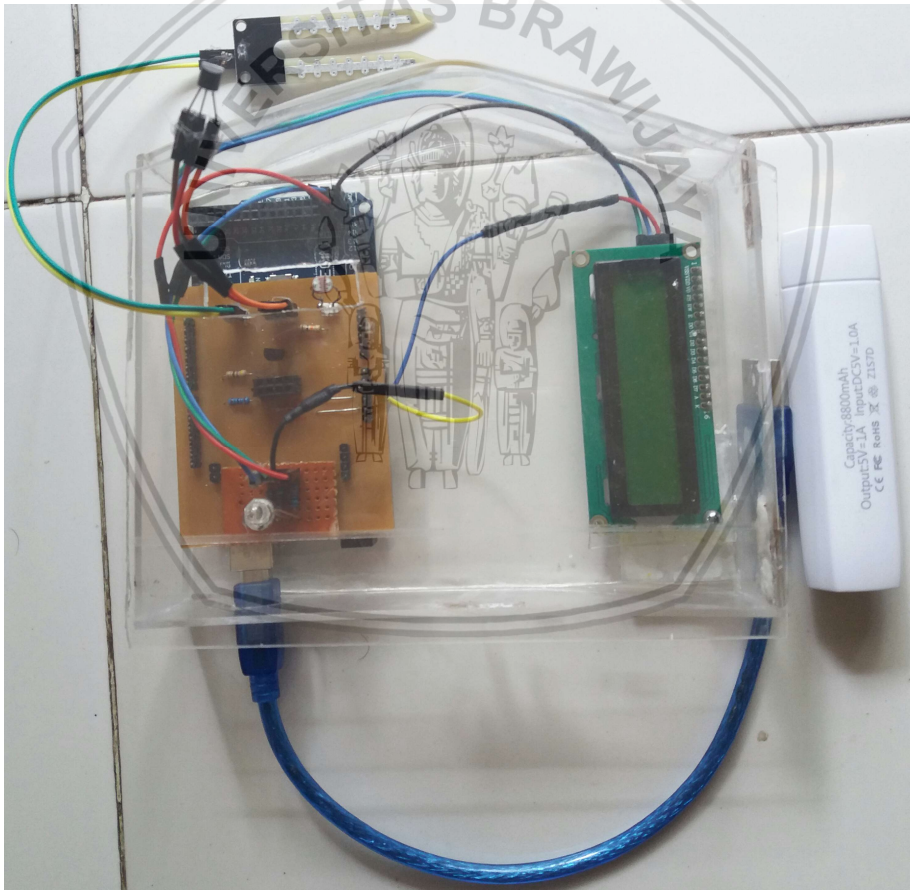
Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe" oleh Sandhy Hermawan.

5.2 Implementasi Sistem

Pada implementasi sistem akan dibahas implementasi perangkat keras dan perangkat lunak berdasarkan perancangan sistem yang telah dibuat. Penjelasan ini mencakup spesifikasi sistem, batasan-batasan implementasi, dan implementasi dar RTOS yang digunakan.

5.2.1 Implementasi Perangkat Keras

Pada tugas akhir ini, implementasi perangkat keras berupa pemasangan komponen Arduino Mega, sensor LDR, sensor LM35, sensor SEN0114, LCD dan LED RGB dalam satu sistem. Pin-pin padaperangkat tersebut dihubungkan sesuai dengan rancangan pada tabel 5.1. Implementasi dari perangkat keras ini berupa prototipe. Implementasi perangkat keras dapat ditunjukan pada Gambar 5.6



Gambar 5.6 Prototipe Sistem

5.2.2 Implementasi Perangkat Lunak

Implementasi perangkat lunak pada tugas akhir merupakan implementasi sistem dari hasil perancangan yang dibuat sebelumnya. Dalam implementasi

perangkat lunak ini direpresentasikan dalam bentuk kode program dalam bahasa C.

5.2.2.1 Implementasi pembuatan *task* pada RTOS

Dalam penerapan *Real Time Operating System* dalam program, langkah pertama yang harus dilakukan adalah mendefinisikan *task-task* yang akan digunakan. Sesuai dengan prinsip RTOS yang digunakan maka didefinisikan 7 *task* yaitu *Task*LDR, *Task*LM35, *Task*Mois, LCD, *Tracehole*LDR, *Tracehole*LM35, dan *Tracehole*Mois. Karena RTOS pada sistem ini menggunakan fungsi *Semaphore* maka fungsi tersebut juga harus dideklarasikan.LM Deklarasi *task* pada RTOS ditampilkan pada Tabel 5.3.

Tabel 5.3 Potongan program definisi *task*

```
void TaskLDR              ( void *pvParameters );
void TaskLM35             ( void *pvParameters );
void TaskMois             ( void *pvParameters );
void LCD                  ( void *pvParametres );
void TraceholeLDR         ( void *pvParameters );
void TraceholeLM35        ( void *pvParameters );
void TraceholeMois        ( void *pvParameters );
SemaphoreHandle_t xSerialSemaphore;
```

Langkah selanjutnya pada bagian void setup() dideklarasikan fungsi *Semaphore* yang bertugas sebagai sumber daya atau *resource*. Ketika *task* yang pertama kali memanggil fungsi *Semaphore* ini maka *task* tersebut akan dijalankan dan *task* lain tidak akan berjalan. Pertama dilakukan pengecekan pada *Semaphore,* jika *Semaphore* kosong maka *Semaphore* akan dibuat. Namun jika *Semaphore* tidak kosong maka *Semaphore* sudah dapat digunakan. Untuk program pembuatan *Semaphore* dapat lihat pada Tabel 5.4.

Tabel 5.4 Potongan program pembuatan *Semaphore*

```
if ( xSerialSemaphore == NULL )
  {
    xSerialSemaphore = xSemaphoreCreateMutex();
    if ( ( xSerialSemaphore ) != NULL )
      xSemaphoreGive( ( xSerialSemaphore ) ); }
```

Setelah sebelumnya *task* sudah dideklarasikan maka selanjutnya adalah membuat *task* dan melakukan setting pada masing-masing *task*. Pembuatan dan setting dari *task* ini meliputi nama, *stack size*, dan *priority*. Untuk nama *task* disesuaikan dengan *task* yang sudah didefinisikan, untuk *stack size* digunakan ukuran *default* yang diberikan oleh FreeRTOS, dan untuk *priority* disesuaikan dengan perancangan sistem yang sudah dibuat. Untuk program pembuatan *task* ditampilkan pada Tabel 5.5.

Tabel 5.5 Potongan program pembuatan *task*

```
xTaskCreate(
    TaskLDR
    ,  (const portCHAR *)"LDR"
    ,  256
    ,  NULL
    ,  3
    ,  NULL );
  xTaskCreate(
    TaskLM35
    ,  (const portCHAR *) "LM35"
    ,  256
    ,  NULL
    ,  3
    ,  NULL );
  xTaskCreate(
    TaskMois
    ,  (const portCHAR *) "Mois"
    ,  256
    ,  NULL
    ,  3
    ,  NULL );
  xTaskCreate(
    LCD
    ,  (const portCHAR *) "LCD"
    ,  256
    ,  NULL
    ,  3
    ,  NULL );
    xTaskCreate(
    TraceholeLDR
    ,  (const portCHAR *)"TraceholeLDR"
    ,  256
    ,  NULL
    ,  2
    ,  NULL );
    xTaskCreate(
    TraceholeLM35
    ,  (const portCHAR *)"TraceholeLM35"
    ,  256
    ,  NULL
```

```
    ,  2
    ,  NULL );
    xTaskCreate(
    TracheleMois
    ,  (const portCHAR *)"TracheleMois"
    ,  256
    ,  NULL
    ,  2
    ,  NULL );
```

Pada potongan program pada Tabel 5.5 pada *Task*LDR dibuat pada *prioritas* ketiga atau *prioritas* tertiggi agar dapat di eksekusi terlebih dahulu, begitu juga dengan *Task*LM35, *Task*Mois dan LCD. Kemudian *Tracehole*LDR, *Tracehole*LM35 dan *Tracehole*Mois dibuat dengan *prioritas* kedua, sehingga setelah *Task*LDR, *Task*LM35, *Task*Mois dan LCD selesai dieksekusi maka akan dilanjutkan dengan *Tracehole*LDR, *Tracehole*LM35 dan *Tracehole*Mois.

5.2.2.2 Implementasi *Task* Pembacaan Sensor Cahaya Suhu dan Kelembaban Tanah pada RTOS

Setelah Setelah *task* sudah dibuat dan disetting maka langkah selanjutnya adalah mengisi *task* tersebut dengan pembacaan sensor LDR. Implementasi program *Task*LDR ditampilkan pada Tabel 5.6.

Tabel 5.6 *Task* pembacaan sensor LDR

```
void TaskLDR(void *pvParameters __attribute__((unused)) )
{
  for (;;)
  {
    if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
    {
    int ldr = analogRead(A3);
    sensor2 = ldr * (5.0 / 1023.0);
    Serial.print("LDR value      = ");
    Serial.print(sensor2);
    Serial.println();

    lcd.setCursor(7,0);
    lcd.print ("ldr");
    lcd.setCursor(7,1);
    lcd.print (sensor2);
    lcd.setCursor(8,1);
    lcd.print (" V ");
```

```
    xSemaphoreGive( xSerialSemaphore );
    }
     vTaskDelay(1000 / portTICK_PERIOD_MS );
  }
}
```

Berdasarkan pada Tabel 5.6 implementasi pada *Task*LDR diawali dengan pengambilan *Semaphore,* lalu memproses hasil sensing LDR pada satuan Volt kemudian hasil perhitungan tersebut ditampilkan pada LCD. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian *delay* pada *task*.

Setelah *Task*LDR selesai dibuat selanjutnya membuat fungsi *Task*LM35. *Task* ini akan berjalan setelah *Task*LDR selesai dieksekusi. *Task*LM35 dijalankan ketika *Task*LDR sudah melepaskan *Semaphore*. Implementasi program *Task*LM35 ditampilkan pada Tabel 5.7.

Tabel 5.7 *Task* pembacaan sensor LM35

```
void TaskLM35(void *pvParameters __attribute__((unused)) )
{
    for (;;)
    {
      if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
      {
      val = analogRead(tempPin);
      float mv = (val/1024.0)*5000;
      sensor1c = mv/10;
      sensor1f = (sensor1c*9)/5 + 32;

      Serial.print("LM35 Value     = ");
      Serial.print(sensor1c);
      Serial.print(" *C ");
      Serial.println();

      lcd.setCursor(0,0);
      lcd.print ("lm");
      lcd.setCursor(0,1);
      lcd.print (sensor1c);
      lcd.setCursor(3,1);
      lcd.print ("*C");

      xSemaphoreGive( xSerialSemaphore );
      }
      vTaskDelay(1000  / portTICK_PERIOD_MS);
```

```
  }
}
```

Berdasarkan pada Tabel 5.7, implementasi pada *Task*LM35 diawali dengan pengambilan *Semaphore*, lalu memproses hasil sensing lm35 pada satuan *C kemudian hasil perhitungan tersebut ditampilkan pada LCD. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian delay pada *task*.

Setelah *Task*LM35 selesai dibuat selanjutnya membuat fungsi *Task*Mois. Pada fungsi *Task*Mois memiliki format *task* yang sama dengan *Task*LDR dan *Task*LM35. *Task* ini akan berjalan setelah *Task*LDR dan *Task*LM35 selesai dieksekusi. *Task*Mois dijalankan ketika *Task*LM35 sudah melepaskan *Semaphore*. Implementasi program *Task*Mois ditampilkan pada Tabel 5.8.

Tabel 5.8 *Task* pembacaan sensor kelembaban tanah

```
void TaskMois(void *pvParameters __attribute__((unused)) )
{
   for (;;)
  {
    if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
    {
    sensor3 = analogRead(moisPin);
    sensor3 = constrain(sensor3, 400, 1023);
    sensor3 = map(sensor3,400,1023,100,0);
    Serial.print("MoisTure value = ");
    Serial.print(sensor3);
    Serial.println(" % ");
    Serial.println();

    lcd.setCursor(11,0);
    lcd.print ("mois");
    lcd.setCursor(11,1);
    lcd.print (sensor3);
    lcd.setCursor(13,1);
    lcd.print (" % ");

    xSemaphoreGive( xSerialSemaphore );
    }
    vTaskDelay(1000 / portTICK_PERIOD_MS);
  }
}
```

Berdasarkan pada Tabel 5.8, implementasi pada *Task*Mois diawali dengan pengambilan *Semaphore*, lalu memproses hasil sensing kelembaban tanah pada

satuan persen kemudian hasil perhitungan tersebut ditampilkan pada LCD. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian *delay* pada *task*.

Setelah *Task*Mois selesai dibuat selanjutnya membuat fungsi LCD. *Task* ini akan berjalan setelah *Task*LDR, *Task*LM35 dan *Task*Mois selesai dieksekusi. LCD dijalankan ketika *Task*Mois sudah melepaskan *Semaphore*. Implementasi program LCD ditampilkan pada Tabel 5.9.

Tabel 5.9 *Task* menghapus layar LCD

```
void LCD (void *pvParameters)
{
  (void) pvParameters;
  for (;;)
  {
    if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) == pdTRUE )
    {

    lcd.clear();

    xSemaphoreGive( xSerialSemaphore );
    }
    vTaskDelay(2000 / portTICK_PERIOD_MS );
  }
}
```

Berdasarkan pada Tabel 5.9, implementasi pada LCD diawali dengan pengambilan *Semaphore*, karena layar LCD menampilkan data sensoring secara terus menerus, maka di perlukan perintah untuk menghapus layar lcd agar sensing data pada tanaman stroberi bisa terus di perbaharui secara *real time*. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian delay pada *task*.

5.2.2.3 Implementasi lampu peringatan LED RGB pada RTOS

Setelah *task* pembacaan sensor sudah dibuat dan disetting makan langkah selanjutnya adalah membuat *task* untuk peringatan kondisi cayaha, suhu dan kelembaban tanah sesuai syarat tumbuh baik pada tanaman stroberi. Implementasi program lampu peringatan pada suhu ditampilkan pada Tabel 5.10.

Tabel 5.10 *Task* lampu peringatan untuk pencahayaan pada tanaman stroberi

```
void TraceholeLDR (void *pvParameters __attribute__((unused)) )
{
    for (;;)
    {
```

```
      if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
      {
        if (sensor2 <= 1)
      {
        digitalWrite(red, HIGH);
        digitalWrite(green, HIGH);
        digitalWrite(blue, LOW);
      }
      else
      {
        digitalWrite(red, HIGH);
        digitalWrite(green, HIGH);
        digitalWrite(blue, HIGH);
      }

      xSemaphoreGive( xSerialSemaphore );
      }
      vTaskDelay(1000  / portTICK_PERIOD_MS);
  }
}
```

Berdasarkan pada Tabel 5.10, implementasi lampu peringatan pencahyaan pada LED RGB diawali dengan pengambilan *Semaphore*, kemudian peneliti membuat kondisi untuk sensing pencahyaan yang sudah di kerjakan oleh *Task*LDR berupa variable sensor2. Peneliti memberi batasan nilai sensing kurang dari 1 V, jika sensing pada pencahyaan sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing pencahyaan tidak sesuai dengan batasan nilai tersebut maka LED BLUE akan menyala itu berarti pencahyaan yang didapat oleh tanaman stroberi tidak terpenuhi. LDR sensor adalah sebuah *resistor* yang nilai resistansi nya tergantung pada cahaya yang jatuh pada permukaannya.

Jika cahaya yang masuk sangat terang sekali, nilai hambatan LDR akan menjadi sangat kecil sekali, sehingga tegangan pada titik ukur mendekati 5V. Sedangkan pada keadaan sangat gelap, hambatan LDR menjadi sangat besar sekali sehingga tegangan di titik ukur mendekati 0V. Dalam kenyataannya peneliti mendapatkan titik tegangan di antara 0V dan 5V sesuai intensitas cahaya yang masuk. Sensor cahaya ini dapat peneliti gunakan untuk pemantauan terhapada pencahyaan sinar matahari pada tanaman stroberi. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian delay pada *task*.

Setelah *Tracehole*LDR selesai dibuat selanjutnya membuat *Tracehole*LM35. *Task* ini akan berjalan setelah *Task*LDR, *Task*LM35, *Task*Mois, LCD, dan *Tracehole*LDR selesai dieksekusi. *Tracehole*LM35 dijalankan ketika *Tracehole*LDR sudah melepaskan *Semaphore*. Implementasi program *Tracehole*LM35 ditampilkan pada Tabel 5.11.

Tabel 5.11 *Task* lampu peringatan untuk suhu pada tanaman stroberi

```
void TracheholeLM35 (void *pvParameters __attribute__((unused)) )
{
    for (;;)
    {
      if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
      {
        if ((sensor1c >= 20) || (sensor1c <= 17))      {
        digitalWrite(red, LOW);
        digitalWrite(green, HIGH);
        digitalWrite(blue, HIGH);
      }
      else
      {
        digitalWrite(red, HIGH);
        digitalWrite(green, HIGH);
        digitalWrite(blue, HIGH);
      }

      xSemaphoreGive( xSerialSemaphore );
      }
      vTaskDelay(1000  / portTICK_PERIOD_MS);
  }
}
```

Berdasarkan pada Tabel 5.11, implementasi lampu peringatan suhu pada LED RGB diawali dengan pengambilan *Semaphore*, kemudian peneliti membuat kondisi untuk sensing suhu yang sudah di kerjakan oleh *Task*LM35 berupa variable sensor1c, lalu peneliti memberi batasan nilai antara 17 *C sampai dengan 20 *C, jika sensing pada suhu sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing suhu tidak sesuai dengan batasan nilai tersebut maka LED RED akan menyala itu berarti suhu yang didapat oleh tanaman stroberi tidak sesuai untuk syarat tumbuh baik tanaman stroberi. Batasan nilai tersebut di dapat dari journal yang di kaji oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp dan journal yang berjudul "Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe" oleh Sandhy Hermawan. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian delay pada *task*.

Setelah *Tracehole*LM35 selesai dibuat selanjutnya membuat *Tracehole*Mois. *Task* ini akan berjalan setelah *Task*LDR, *Task*LM35, *Task*Mois, LCD, *Tracehole*LDR dan *Tracehole*LM35 selesai dieksekusi. *Tracehole*Mois dijalankan

ketika *Tracehole*LM35 sudah melepaskan *Semaphore*. Implementasi program *Tracehole*Mois ditampilkan pada Tabel 5.12.

Tabel 5.12 *Task* lampu peringatan untuk kelembaban tanah pada tanaman stroberi

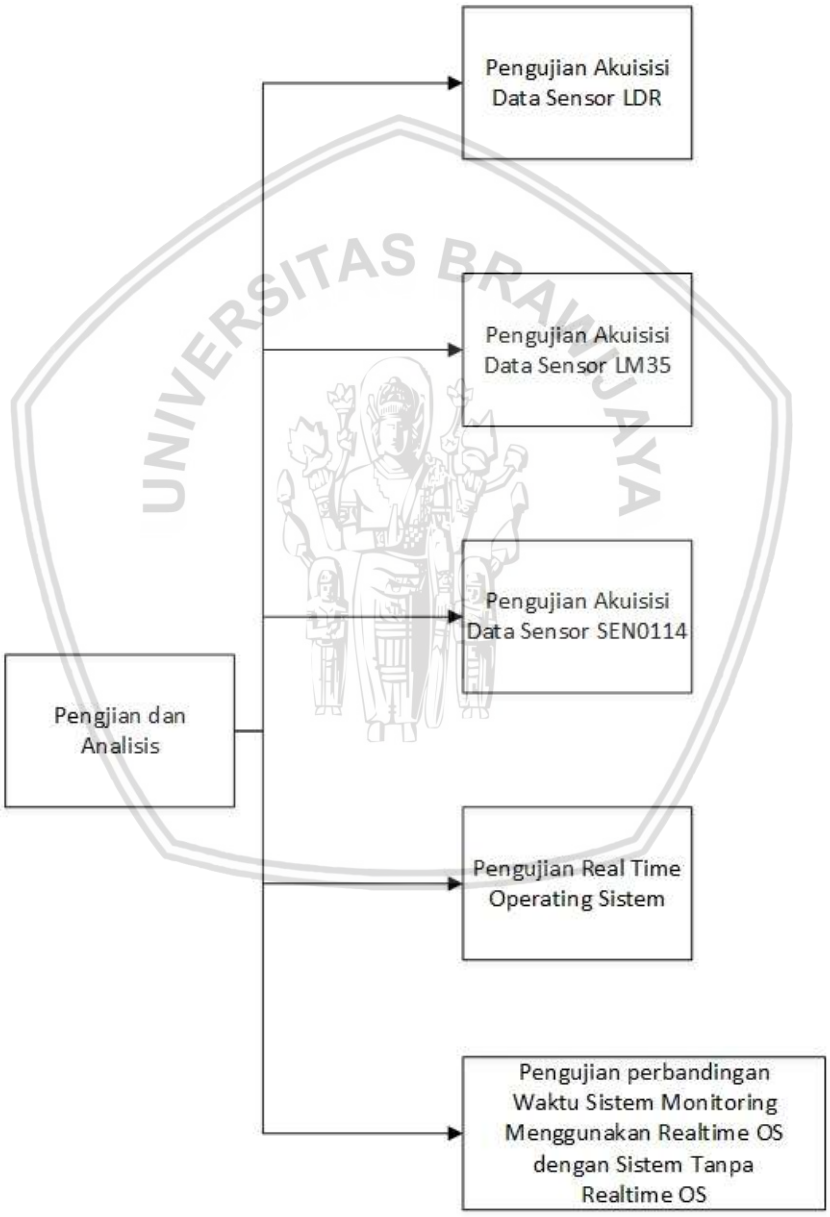
```
void TracheholeMois (void *pvParameters __attribute__((unused)) )
{
    for (;;)
    {
      if ( xSemaphoreTake( xSerialSemaphore, ( TickType_t ) 1000 ) ==
pdTRUE )
      {
        if ((sensor3 >= 90) || (sensor3 <= 80)) // if (  20 < sensor1
<17 )
      {
        digitalWrite(red, HIGH);
        digitalWrite(green, LOW);
        digitalWrite(blue, HIGH);
      }
      else
      {
        digitalWrite(red, HIGH);
        digitalWrite(green, HIGH);
        digitalWrite(blue, HIGH);
      }
      xSemaphoreGive( xSerialSemaphore );
      }
      vTaskDelay(1000  / portTICK_PERIOD_MS);
  }
}
```

Berdasarkan pada Tabel 5.12, implementasi lampu peringatan kelembaban tanah pada LED RGB diawali dengan pengambilan *Semaphore*, kemudian peneliti membuat kondisi untuk sensing kelembaban tanah yang sudah di kerjakan oleh *Task*Mois berupa variable sensor3, lalu peneliti memberi batasan nilai antara 80 sampai dengan 90% sesuai syarat tumbuh baik tanman stroberi, jika sensing pada kelembaban tanah sesuai dengan batasan nilai tersebut maka LED RGB mati, tapi jika sensing kelembaban tanah tidak sesuai dengan batasan nilai tersebut maka LED GREEN akan menyala itu berarti kelembaban tanah yang didapat oleh tanaman stroberi tidak sesuai untuk syarat tumbuh baik tanaman stroberi. Batasan nilai tersebut di dapat dari journal yang di kaji oleh Kantor Deputi Menegristek Bidang Pendayagunaan dan Pemasyarakatan Ilmu Pengetahuan dan Teknologi, MIG corp dan journal yang berjudul "Kajian Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman

Fungsional Stroberi Jahe" oleh Sandhy Hermawan. Selanjutnya *task* akan diakhiri dengan pelepasan *Semaphore* dan pemberian *delay* pada *task*.

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab ini membahas mengenai proses pengujian sistem monitoring tanaman stroberi. Terdapat 4 pengujian dalam sistem ini, yaitu pengujian akusisi data yang terdiri dari pembacaan sensor LDR, LM35 dan SEN0114, pengujian indikator LED RGB, pengujian RTOS, dan pengujian perbedaan sistem yang menggunakan RTOS dengan sistem tanpa menggunakan RTOS. Hal tersebut ditunjukan pada Gambar 6.1.



Gambar 6.1 Pohon Pengujian dan Analisis

6.1 Pengujian Akuisisi Data sensor LDR

6.1.1 Tujuan

Untuk menguji sensitivitas sensor LDR dalam membaca pencahayaan pada tanaman stroberi. Dimana output dari sensor LDR sudah mempresentasikan batasan nilai intensitas cahaya untuk tanaman stroberi yaitu kurang dari 1V.

6.1.2 Prosedur

Prosedur pengujian sensor LDR ini meliputi:

1. Sambungkan sensor LDR dengan mikrokonroller Arduino Mega.
2. Buka Software Arduino IDE, lalu compile dan upload source code program untuk LDR.
3. Buka serial monitor pada Arduino IDE, amati output dari sensor LDR. Apabila output belum stabil maka tunggu beberapa waktu hingga sensor stabil. Setelah stabil sensor siap untuk melakukan sensing.
4. Untuk memberikan nilai masukan pada sensor, penulis menempatkan sensor di ruang tertutup, ruang tertutup dengan diberikan senter dan di ruang terbuka untuk meningkatkan pembacaan intensitas cahaya dari LDR.
5. Amati hasil melalui serial monitor dan ambil setiap sampel lalu bandingkan hasil pembacaan intensitas cahaya tersebut dengan batasan nilai pencahayaan untuk tanaman stroberi.
6. Kesimpulan

6.1.3 Hasil dan Analisis

Tabel 6.1 Hasil Pengujian Sensor LDR Secara *Real Time*

Nilai Pembacaan Sensor (Volt)	Batasan nilai <1 Volt untuk LED Biru	Kondisi
0	Menyala	Ruang Tertutup
0		
0		
2	Mati	Ruang tertutup yang diberikan input berupa senter dengan jarak 30 cm
2		
2		
2		
2		

4	Mati	Ruang terbuka dengan disinari matahari secara langsung
4		
4		

Pada tabel 6.1 saat sensor LDR di tempatkan di ruang tertutup dengan intensitas cahaya yang kurang, sensing pada sensor LDR bernilai 0 Volt dan LED Biru menyala karena kondisi tersebut tidak memenuhi batasan nilai yang baik untuk tanaman stroberi, dapat dikatakan kondisi tersebut sensor belum mendeteksi adanya intensitas cahaya. Kemudian kondisi di saat sensor LDR di berikan input berupa nyala senter dengan jarak 30cm sensing pada sensor LDR bernilai 2 Volt dan lampu LED Biru mati, berarti sensor mendeteksi intensitas cahaya pada senter. Pada tahap selanjutnya peneliti menempatkan sensor di ruang terbuka pada siang hari yang cerah, ketika sensor di tempatkan di ruang terbuka sensing pada sensor LDR bernilai 4 Volt dan lampu LED Biru mati, itu dikarenakan nilai tersebut sudah memenuhi batasan nilai untuk sistem monitoring tanaman stroberi. Dari pengujian diatas dapat disimpulkan sensor berhasil mendeteksi intensitas cahaya yang di eksekusi secara *real time* dan lampu peringatan berjalan sesuai batasan nilai yang di tentukan.

6.2 Pengujian Akuisisi Data sensor LM35

6.2.1 Tujuan

Untuk menguji sensitivitas sensor LM35 dalam membaca suhu lingkungan pada tanaman stroberi. Dimana output dari sensor LM35 sudah mempresentasikan batasan nilai suhu yang baik untuk syarat tumbuh tanaman stroberi yaitu 17*C sampai dengan 20*C.

6.2.2 Prosedur

Prosedur pengujian sensor LM35 ini meliputi:

1. Sambungkan sensor LM35 dengan mikrokonroller Arduino Mega.
2. Buka Software Arduino IDE, lalu compile dan upload source code program untuk LM35.
3. Buka serial monitor pada Arduino IDE, amati output dari sensor LM35. Apabila output belum stabil maka tunggu beberapa waktu hingga sensor stabil. Setelah stabil sensor siap untuk melakukan sensing.
4. Untuk memberikan nilai masukan pada sensor, penulis menggunakan uap dari air panas untuk meningkatkan suhu dari LM35.
5. Amati hasil melalui serial monitor dan ambil 10 sampel lalu bandingkan hasil pembacaan suhu tersebut dengan batasan nilai suhu yang baik untuk syarat tumbuh tanaman stroberi.
6. Kesimpulan

6.2.3 Hasil dan Analisis

Tabel 6.2 Hasil Pengujian Sensor LM35 Secara *Real Time*

Nilai Pembacaan Sensor (Celcius)	Batasan nilai 17-20*C untuk LED Merah	Kondisi
27	Menyala	Kondisi Normal
27		
27		
27		
27		
29	Menyala	Setelah diberi uap air panas terhadap sensor
30		
31		
32		
31		

Berdasarkan table 6.2 saat source code untuk LM35 selesai diupload, pembacaan sensor langsung menunjukan nilai 27*C, itu berarti suhu ruangan pada saat prototype dijalankan ± 27*C dan LED merah menyala karena nilai tersebut tidak memenuhi batasa nilai yang telah di tentukan untuk suhu yang baik pada tanaman stroberi. Begitu juga saat sensor di beri masukan berupa uap air panas, sensor langsung menunjukan nilai kenaikan menjadi ±31 *C, kenaikan nilai tersebut di sebabkan sensor membaca uap panas pada air tersebut ,LED merah tetap menyala sebab, nilai tersebut juga tidak memenuhi batasan nilai untuk suhu yang baik pada tanaman stroberi. Dari pengujian di atas dapat disimpulkan sensor berhasil mendeteksi suhu ruangan yang di eksekusi secara real time dan lampu peringatan berjalan sesuai batasan nilai yang ditentukan.

6.3 Pengujian Akuisisi Data sensor SEN0114

6.3.1 Tujuan

Untuk menguji sensitivitas sensor SEN0114 dalam membaca kelembaban tanah pada tanaman stroberi. Dimana output dari sensor SEN0114 sudah mempresentasikan batasan nilai kelembaban tanah yang baik untuk syarat tumbuh tanaman stroberi yaitu 80 sampai dengan 90%.

6.3.2 Prosedur

Prosedur pengujian sensor SEN0114 ini meliputi:

1. Sambungkan sensor SEN0114 dengan mikrokonroller Arduino Mega.

2. Buka Software Arduino IDE, lalu compile dan upload source code program untuk SEN0114.
3. Buka serial monitor pada Arduino IDE, amati output dari sensor SEN0114. Apabila output belum stabil maka tunggu beberapa waktu hingga sensor stabil. Setelah stabil sensor siap untuk melakukan sensing.
4. Untuk memberikan nilai masukan pada sensor, penulis menggunakan air yang dituangkan pada pot tanaman stroberi untuk meningkatkan kelembaban tanah dari SEN0114.
5. Amati hasil melalui serial monitor dan ambil 3 sampel lalu bandingkan hasil pembacaan kelembaban tanah tersebut dengan batasan nilai kelembaban tanah yang baik untuk syarat tumbuh tanaman stroberi.
6. Kesimpulan

6.3.3 Hasil dan Analisis

Tabel 6.3 Hasil Pengujian Sensor SEN0114 Secara *Real Time*

Nilai Pembacaan Sensor (%)	Batasan nilai 80-90 untuk LED Hijau	Kondisi
0	Menyala	Kondisi Awal Sebelum Sensor di tancapkan kedalam tanah
0		
17	Menyala	Setelah sensor di tancapkan ke dalam tanah
17		
17		
40	Menyala	Setelah pot di siram air sebanyak 150ml
40		
39		
87	Mati	Setelah pot di siram air sebanyak 500ml
86		
86		
92	Menyala	Setelah pot di siram air sebanyak 1500ml
92		
92		

Berdasarkan table 6.2 saat sensor SEN0114 belum tertancap pada tanah sensor bernilai 0% dan LED hijau menyala, ini kondisi pembacaan sensor pada udara. Kemudian ketika sensor di tancapkan kedalam tanah pembacaan sensor menjadi 17%, lalu tahap selanjutnya peneliti menambahkan air sebanyak 150ml kedalam pot. Setelah pot di isi air, pembacaan sensor langsung naik menjadi 39-

40% tetapi LED hijau masih menyala karena nilai tersebut belum memenuhi batasan nilai untuk tanaman stroberi. Untuk tahap selanjutnya peneliti menunggu sekitar 30 menit untuk kelembaban tanah kembali menjadi 17%.

Tahap kedua setelah menunggu sekitar 30 menit dan kelembaban tanah sudah kembali menjadi 15-17%, peneliti menambahkan air sebanyak 500ml kedalam pot. Setelat pot di isi air sebanyak 500ml, pembacaan sensor naik menjadi 86-87% dan LED hijau mati, karena nilai tersebut sudah memenuhi syarat kelembaban tanah untuk stroberi yaitu 80-90%. Untuk tahap terakhir peneliti menunggu sampai kelembaban tanah kembali menjadi 15-17%.

Tahap terakhir setelah menunggu cukup lama sampai sensor kembali membaca kelembaban tanah pada kisaran 15-17%, peneliti menambahkan air sebanyak 1500ml kedalam pot. Setelah pot di isi air sebanyak 1500ml, terjadi kenaikan secara dratis pembacaan sensor dari 15% ke 92-97% dan LED hijau menyala, itu di karenakan kondisi tersebut terlalu lembab dan tidak memenuhi syarat kelembaban tanah untuk stroberi. Dari pengujian diatas dapat disimpulkan sensor berhasil mendeteksi kadar air yang dimasukan ke dalam tanah kemudian di eksekusi secara real time dan lampu peringatan berjalan sesuai batasan nilai yang di tentukan.

6.4 Pengujian *Real-Time Operating System* (RTOS)

Pada pengujian ini meliputi pengujian eksekusi *task* berdasarkan *prioritas*, pengujian waktu eksekusi tiap *task* dan pengujian total waktu eksekusi dari semua *task* yang berjalan pada RTOS. Pengujian ini dilakukan karena RTOS pada sistem ini bertugas sebagai scheduler dimana *prioritas* dan waktu eksekusi dari sistem sangat diperhitungkan. Berikut pengujiannya.

6.4.1 Pengujian urutan eksekusi *task* berdasarkan *prioritas*

6.4.1.1 Tujuan

Untuk memastikan eksekusi *task* pada program yang dibuat sesuai dengan *prioritas* yang sudah ditentukan pada tabel 5.2.

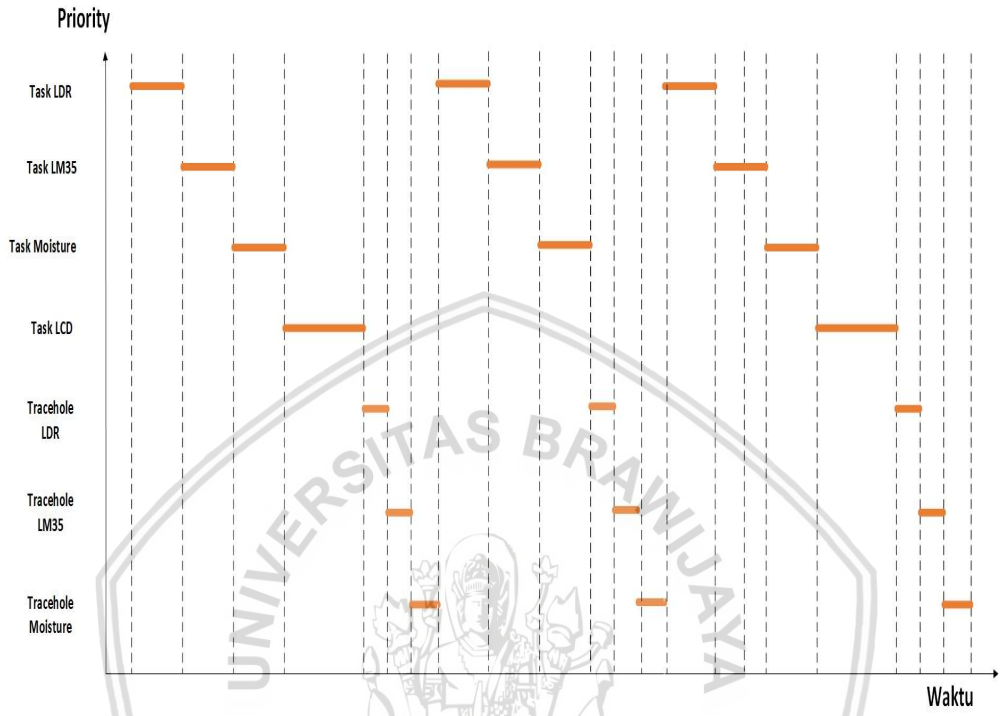
6.4.1.2 Prosedur

Pada pengujian ini akan diuji *prioritas* dari sistem, dimana *prioritas* ini bertugas untuk menentukan *task* mana yang akan di eksekusi terlebih dahulu. Prosedur pengujian eksekusi *task* berdasarkan *prioritas* meliputi:

1. Sambungkan prototipe alat dengan komputer atau laptop.
2. Buka Software Arduino IDE, setelah itu buka source code alat yang dibuat dan setelah itu compile dan upload source code programnya.
3. Amati hasil melalui serial monitor.
4. Kesimpulan.

6.4.1.3 Hasil dan Analisis

Hasil dari pengujian urutan eksekusi *task* pada sistem ditampilkan pada Gambar 6.2



Gambar 6.2 Analisis hasil pengujian eksekusi *task* berdasarkan *prioritas*

Berdasarkan analisis Gambar 6.2, *Task* LDR yang memiliki *prioritas* terbesar di eksekusi terlebih dahulu, hal ini dikarenakan pada *task* LDR diberi *prioritas* 3 dimana, *prioritas* tersebut merupakan *prioritas* terbesar dalam sistem. Selama *task* 3 dieksekusi *task* dengan *prioritas* yang berada dibawahnya akan menunggu di eksekusi sampai *task* 3 menyelesaikan tugasnya, hal ini dikarenakan terdapat fungsi *Semaphore*() pada sistem yang bertugas agar *task* tersebut tidak dapat di *interrupt* oleh *task* lain selama *task* tersebut masih dieksekusi. Ketika *task* LDR selesai dieksekusi maka *task* LM35, *task* Moisture, *task* LCD, *Tracehole* LDR, *Tracehole* LM35 dan *Tracehole* Moisture akan di eksekusi sesuai urutan *prioritas*.

6.4.2 Pengujian waktu eksekusi *task* pada sistem

6.4.2.1 Tujuan

Untuk memastikan waktu eksekusi setiap *task* dan semua *task* pada program tidak melebihi *deadline* yang ditentukan.

6.4.2.2 Prosedur

Untuk membantu perhitungan waktu eksekusi tiap *task* digunakan fungsi timer pada arduino yaitu fungsi “micros()” untuk menghitung waktu eksekusi dalam satuan microsecond dan “millis()” untuk menghitung waktu eksekusi dalam

satuan milisecond. Fungsi micros() dan millis() akan diletakan pada awal dan akhir program dari setiap *task* sesudah saat *Semaphore* diambil dan sebelum *Semaphore* dilepas untuk mendapatkan waktu dari eksekusi *task*. Prosedur pengujian waktu eksekusi *task* meliputi:

1. Sambungkan prototipe alat dengan komputer atau laptop.
2. Buka Software Arduino IDE, setelah itu compile dan upload source code programnya.
3. Amati hasil melalui serial monitor.
4. Kesimpulan.

6.4.2.3 Hasil dan Analisis

Hasil pengujian waktu eksekusi tiap *task* pada sistem ditampilkan pada Tabel 6.5.

Tabel 6.4 Analisis hasil pengujian waktu eksekusi tiap *task*

Task	**Waktu Eksekusi**	***Deadline***	**Hasil**
Task LDR	± 14 ms	1000 ms	Tidak Melebihi *Deadline*
Task Suhu	± 15,4 ms		Tidak Melebihi *Deadline*
Task Mois	± 18,5 ms		Tidak Melebihi *Deadline*
Task LCD	± 3,5 ms	2000 ms	Tidak Melebihi *Deadline*
Tracehole LDR	± 0,02 ms	1000 ms	Tidak Melebihi *Deadline*
Tracehole LM35	± 0,02 ms		Tidak Melebihi *Deadline*
Tracehole Mois	± 0,02 ms		Tidak Melebihi *Deadline*
Total Waktu	**± 89 ms**		Tidak Melebihi *Deadline*

Berdasarkan hasil analisis yang terdapat pada tabel 6.4 dapat diketahui bahwa waktu eksekusi dari setiap *task* tidak melebihi *deadline* waktu yang sudah ditentukan. *Task* LDR, *Task* Suhu dan *Task* Mois memiliki waktu eksekusi lebih besar dari *task* lain, hal ini di karenakan pada *task-task* tersebut terdapat banyak perhitungan serta terdapat fungsi untuk menampilkan nilai sensoring pada LCD. Pada *Task* LCD, *Tracehole* LDR, *Tracehole* LM35, dan *Tracehole* Mois tidak memerlukan banyak waktu dikarenakan pada *task* ini hanya terdapat perhitungan sederhana.

6.5 Pengujian Perbandingan Waktu Eksekusi Sistem *Monitoring* Menggunakan *Real-Time Operating System* (RTOS) dan tanpa RTOS

6.5.1 Tujuan

Untuk membandingkan sistem yang menggunakan RTOS dengan sistem tanpa menggunakan RTOS dari segi kecepatan eksekusi *task*.

6.5.2 Prosedur

Dalam pengujian ini sudah dibuat source code sederhana yang sama namun tidak menggunakan RTOS. Untuk membantu perhitungan waktu eksekusi tiap *task* digunakan fungsi timer pada arduino yaitu fungsi micros() dan millis(), fungsi ini digunakan untuk menghitung waktu dalam satuan microsecond dan milisecond. Prosedur pengujian waktu eksekusi *task* meliputi:

1. Sambungkan prototipe alat dengan komputer atau laptop.
2. Buka Software Arduino IDE, setelah itu buka source code sistem dengan RTOS dan tanpa RTOS. Setelah itu compile dan upload source code programnya.
3. Amati hasil melalui serial monitor dan ambil 10 sampel dari masing-masing sistem.
4. Bandingkan hasil waktu eksekusi sistem dengan RTOS dan tanpa RTOS.
5. Kesimpulan.

6.5.3 Hasil dan Analisis

Hasil pengujian perbandingan waktu eksekusi dari kedua sistem ditampilkan pada Tabel 6.5.

Tabel 6.5 Hasil perbandingan waktu eksekusi sistem dengan RTOS dan tanpa RTOS

Sample	Waktu eksekusi dengan RTOS (ms)	Waktu eksekusi Tanpa RTOS(ms)
1	77	77
2	89	76
3	89	77
4	90	78
5	89	77
6	89	77
7	93	77
8	90	77
9	89	76
10	89	77
Rata - rata	88,4	76.9

Bedasarkan hasil pengujian pada Tabel 6.5, diketehui bahwa sistem yang menggunakan RTOS memerlukan waktu eksekusi lebih lama dibandingkan sistem yang tanpa menggunakan RTOS. Hal ini dapat dilihat bahwa selisih waktu eksekusi dari sistem sekitar ± 11,5 ms. Hal ini dapat disebabkan oleh beberapa faktor salah

satunya waktu perpindahan ketika *Semaphore* dilepas dari satu *task* dan diterima oleh *task* lainnya. Pada sistem ini watu perpindahan tersebut memerlukan 1,64ms.

BAB 7 PENUTUP

Bab ini berisi kesimpulan yang didasarkan dari pengujian dan analisis yang dilakukan selama proses penelitian dan saran yang berisi hal-hal yang diperlukan dalam melakukan pengembangan untuk topik skripsi selanjutnya.

7.1 Kesimpulan

Berdasarkan pengujian dan analisa yang dilakukan teradap tugas akhir ini maka didapatkan kesimpulan sebagai berikut:

1. Proses akusisi data pada LM35 meliputi pembacaan suhu yang terdeteksi di sekitar, pada akuisisi data pada LDR meliputi pembacaan intensitas cahaya yang didapatkan oleh tanaman stroberi, sedangkan pada SEN0114 meliputi pembacaan kelembaban tanah pada tanaman stroberi.
2. Penentuan *prioritas* dan waktu *deadline* pada setiap *task* (RTOS) berpengaruh terhadap urutan *task* yang akan dieksekusi. Apabila penentuan *prioritas* dan waktu pada *task* tidak seimbang maka urutan *task* yang dieksekusi tidak bisa diprediksi.
3. RTOS pada sistem berpengaruh sebagai penentu *deadline* tiap *task*. Apabila *task* yang di eksekusi melebihi *deadline* yang diberikan maka sistem tidak akan berjalan. Berdasarkan hasil pengujian waktu eksekusi tiap *task* dan total semua *task* tidak melebihi *deadline* yang diberikan.
4. Dari segi kecepatan waktu eksekusi sistem menggunakan RTOS memerlukan waktu lebih lama dibandingkan sistem tanpa RTOS. Waktu eksekusi sistem RTOS untuk menjalankan semua *task* adalah ± 88,4ms dan waktu eksekusi sisten yang tanpa RTOS adalah 76,9ms. Hal ini dikarenakan pada sistem RTOS terdapat kegiatan terima-lepas *Semaphore* dari satu *task* ke *task* lainnya, dimana kegiatan memerlukan watu 1,64ms.

7.2 Saran

1. Diharapkan pada penelitian selanjutnya dapat ditambahkan fitur monitoring tanaman stroberi.
2. Diharapkan juga pada penelitian berikutnya dapat lebih memanfaatkan fitur yang ada pada RTOS agar sistem dapat bekerja lebih baik.

DAFTAR PUSTAKA

Anh & Tan, 2009. *Generic Operating Systems Versus RTOS.* Singapura: Nanyang Technological University

Arduino, t.thn. *Arduino Mega Board.* [online] Tersedia di: http://arduino.cc/en/Main/arduinoBoardMega [Diakses 10 Juli 2017].

Ariyanto, E., 2010. *Sistem Operasi Waktu Nyata*. Bandung: Institute Teknologi Telkom.

Banzi, M., 2008. *Getting Started with Arduino*. Inggris: O'Reilly.

Barry, R., 2016. *Mastering the FreeRTOS™ Real Time Kernel*. [Online] Tersedia di: http://www.FreeRTOS.org [Diakses 28 Agustus 2017].

Budiman & Saraswati, 2010. *Berkebun Stroberi Secara Komersial*. Jakarta: Penebar Swadaya..

Cahyono, B., 2011. *Sukses Budi Daya Stroberi di Pot & Perkebunan*. Bandung: Agromedia Pustaka.

Casely & Kumar, 1987. *Project Monitoring and Evaluation in Agriculture.*. s.l.:s.n.

Clayton & Petry, 1983. *Monitoring for Agricultural and Rural Development*. London: The Macmillan.

Darwis, S., 2007. *Sejarah Tanaman Stroberi*. Jakarta: Pusat Penelitian dan Pengembangan Tanaman Industri.

Departemen Pertanian, 2014. *Penurunan Jumlah Produksi Buah Stroberi*. [online] Tersedia di: http://repository.usu.ac.id/bitstream/123456789/44665/5 [Diakses 19 Juli 2017].

Dewi, Y., 2013. *Karakteristik Buah Stroberi dan Khasiatnya*. [online] Tersedia di: http://buah-strawberry.blogspot.co.id/2013/05/tentang-buah-strawberry.html [Diakses 1 Agustus 2017].

Hermawan, S., 2016. Kajian *Perbandingan Stroberi Dengan Ekstrak Jahe dan Konsentrasi Penstabil Terhadap Karakteristik Minuman Fungsional Stroberi Jahe*. Bandung: Universitas Pasundan.

Jatmiko, W., 2015. *RTOS Teori dan Aplikasi*. Depok: Fakultas Ilmu Komputer Universitas Indonesia.

Kumorotomo, W., 2003. *Konsep Dasar Pemantauan dan Evaluasi*. Yogyakarta: Universitas Gajah Mada.

Kurnia, A., 2005. *Petunjuk Praktis Budi Daya Stroberi*. Jakarta: Agromedia Pustaka.

Kusrini, 2008. *Konsep dan Aplikasi Sistem Pendukung Keputusan*. Yogyakarta: ANDI.

Laplante, P. A., 2008. *RealTime System Design and Analysis*. Italy: Wiley-IEEE Press.

Mustakini, J., 2008. *Analisis dan Desain Sistem Informasi.* Yogyakarta: ANDI.

Oxfam, 1995. *Monitoring & Evaluasi Parsiatif.* England: University of Humberside.

Ramdani, M., Rakhmatsyah, A. & Anggis, N., 2015. *Plant Monitoring System Using Zigbee and M2M Platform.* Bandung: Universitas Telkom.

Setiani, A., 2007. *Budidaya dan Analisis Usaha Stroberi*. Jakarta: Sinar Cemerlang Abadi.

Soemadi, W., 1997. *Budidaya Stroberi Di Pot dan Di Kebun..* Solo: CV. Aneka.